INŻYNIERIA BEZPIECZEŃSTWA

LABORATORIUM NR 2

ALGORYTM XOR
ŁAMANIE ALGORYTMU XOR

1. Algorytm XOR

Operacja XOR to inaczej alternatywa wykluczająca, oznaczona symbolem „^" w języku C i symbolem ⊕ w matematyce. Operacja ta działa na wartościach binarnych, zdefiniowana jest następująco:

⊕	0	1
0	0	1
1	1	0

Właściwości algorytmu XOR:

$$a \oplus a = 0 \quad (1)$$
$$a \oplus b \oplus a = b \quad (2)$$

Algorytm XOR nie jest niczym więcej, niż szyfrem Vigenere'a. Jest on do dziś stosowany w komercyjnych pakietach oprogramowania, zwłaszcza przeznaczonego dla systemu Windows, i reklamowany jest jako „prawie tak bezpieczny, jak DES". Jak zostanie za chwilę pokazane, algorytm ten nie zapewnia praktycznie żadnego bezpieczeństwa. W USA algorytm ten jest stosowany z kluczem 160-bitowym w urządzeniach telefonii komórkowej do szyfrowania rozmów na terenie całego kraju.

Algorytm XOR jest algorytmem symetrycznym. Kryptogram jest wynikiem binarnego sumowania modulo 2 tekstu jawnego z kluczem. Jedna procedura, realizująca algorytm XOR, może być zastosowana zarówno do operacji szyfrowania, jak i deszyfrowania, ponieważ dwukrotne sumowanie modulo 2 tej samej wartości daje w wyniku tę wartość:

$$M \oplus K = C$$
$$C \oplus K = M$$

2. Łamanie algorytmu XOR

Algorytm XOR można złamać bardzo łatwo, jednakże łamanie ograniczone jest kilkoma warunkami:
– klucz szyfrujący musi być użyty cyklicznie;
- długość kryptogramu musi być wystarczająca;

- kryptoanalityk musi znać typ zaszyfrowanych danych (tekst, plik BMP itp).

W niniejszej instrukcji ograniczamy się do opisania sposobu łamania kryptogramów z zaszyfrowanym tekstem w języku polskim. Ograniczona jest również długość klucza, który będzie mógł zostać znaleziony.

Instrukcja postępowania:

1. Stworzyć tablicę, zawierającą dwie kolumny: Przesunięcie i Ilość. Tablicę wypełnić w sposób następujący: do kolumny Przesunięcie wpisać kolejne liczby od 1 do zakładanej maksymalnej długości klucza (można założyć np 200); kolumnę Ilość wypełnić zerami.

2. Uzupełnić dane w kolumnie Ilość: należy dla każdego przesunięcia (wpisanego w kolumnę Przesunięcie w tablicy) obliczyć ile znaków jest takich samych. Dla przesunięcia 1 porównujemy znaki nr 1 i 2, potem 2 i 3 itd. Dla przesunięcia 3 porównujemy znaki nr 1 i 4, potem 2 i 5, 3 i 6 itd.

3. Posortować dane w tablicy malejąco, pod względem kolumny Ilość.

4. Obliczyć długość klucza: należy wziąć trzy wartości przesunięć, odpowiadające największej ilości tych samych znaków (czyli wartości z kolumny Przesunięcie z dwóch górnych wierszy tablicy), i wyliczyć ich NWD (Największy Wspólny Dzielnik), np za pomocą algorytmu Euklidesa. Wyliczona wartość jest długością klucza.
   Uwaga. W przypadku, gdy długość klucza jest równa 1, można spróbować zamiast trzech wartości wziąć dwie. Jeśli długość klucza nadal będzie równa 1, należy założyć, że klucz rzeczywiście ma 1 znak (aczkolwiek możliwe jest, że analizowane dane nie są kryptogramem).

5. Znaki klucza mogą być całkowicie dowolne (z całego zakresu ASCII), dlatego też należy sprawdzić wszystkie. W tym celu należy utworzyć drugą tablicę, zawierającą dwie kolumny: ASCII oraz Ilość. Kolumnę ASCII należy wypełnić liczbami od 0 do 255 (zakres ASCII), a kolumnę Ilość wypełnić ją zerami. Należy założyć też zbiór znaków, które występują najczęściej; zwykle zakłada się wszystkie widzialne znaki (spacja, duże i małe litery, cyfry).

   Uwaga. Zbiór ten można zmienić (np ograniczyć tylko do występujących najczęściej znaków – ‘ ‘, ‘a’, ‘e’, ‘t’, ‘s’). Zbiór ten ma znaczący wpływ na efekty poszukiwania znaków klucza, dlatego należy dobrać go empirycznie.

6. Przystąpić do wyznaczania samego klucza. W tym celu należy z odpowiednim przesunięciem (równym długości klucza) dokonywać operacji XOR na znakach kryptogramu i wszystkich możliwych znakach kodu ASCII, symbolizujących literę klucza. Na przykład dla klucza o długości 10 szukając pierwszego znaku przetwarzamy co 10 znak kryptogramu (znaki nr 1, 11, 21 itd) sumując każdy przetwarzany znak kryptogramu modulo 2 z kolejnymi kodami ASCII. Jeśli wynik operacji należy do założonego zbioru znaków, należy zwiększyć o 1 wartość w tablicy

w kolumnie Indeks, dla której wartość w kolumnie ASCII odpowiada drugiemu czynnikowi operacji XOR, czyli znakowi klucza.

Przykład.

Rozważmy kryptogram XOR w następującej postaci (podano kody ASCII znaków w formie heksadecymalnej):

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	25	0D	43	1B	03	10	1B	1B	05	13	0D	14	00	0C	0A	00
2	42	1B	0E	10	4E	04	0F	43	08	42	13	13	0D	01	18	42
3	19	0D	03	0E	00	0C	0A	00	42	17	04	05	0C	41	18	0D
4	00	09	0C	0C	0B	17	04	05	0C	41	03	0F	06	0D	11	18
5	16	0E	14	6F	69											

Tekst ma długość 69 bajtów.

Tworzymy tablicę przesunięć i wypełniamy ją – np są trzy pary identycznych znaków dla przesunięcia o 1 znak: 1B w pierwszej linii, pozycje 7 i 8; 13 w drugiej linii, pozycje 11 i 12; 0C w czwartej linii, pozycje 3 i 4. Podobnie są trzy pary dla przesunięcia o 3 znaki: 1B w pierwszej linii na pozycjach 4 i 7; 0 w pierwszej linii na pozycjach 13 i 16; 0 w trzeciej linii na pozycjach 5 i 8.

Przesunięcie	Ilość
1	3
2	0
3	3
4	1
5	2
6	3
7	2
8	0
9	5
10	1
11	1
12	6

Sortujemy tablicę:

Przesunięcie	Ilość
12	6
9	5
1	3
3	3
6	3
5	2
7	2
4	1
10	1

11	1
2	0
8	0

Na podstawie tabeli posortowanej można ustalić długość klucza. Pierwsze trzy przesunięcia to 12, 9 i 1, a NWD(12,9,1) = 1. Jednak biorąc tylko dwa przesunięcia, otrzymujemy wynik 3 (NWD(12,9) = 3) i powinniśmy wziąć pod uwagę właśnie to rozwiązanie, zwłaszcza, że dalej w tabeli są wartości będące wielokrotnościami 3 – 3 i 6.

Zakładamy zatem, że **klucz ma 3 znaki**.

Szukamy pierwszej litery klucza. Wiedząc, że klucz ma trzy znaki, pierwsza litera klucza była użyta do szyfrowania znaku nr 1, 4, 7, 10 itd. – i tylko te znaki będziemy badać. Szukając drugiej litery klucza, badamy znaki nr 2, 5, 8, 11 itd. Dla kolejnych – analogicznie.
Tworzymy drugą tablicę, zawierającą kody ASCII (szesnastkowo) i częstości wystąpień (dziesiętnie) i wypełniamy ją, szukając pierwszej litery klucza. Wypełniamy następująco: wiersze A_x oznaczają kody ASCII, a wiersze I_x ilość znaków, stanowiących wynik operacji XOR znaku kryptogramu z kolejnymi możliwymi znakami, które mogą wystąpić w kluczu (od 0 do 255).

A_0	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
I_0	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0
A_1	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F
I_1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A_2	20	21	22	23	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F
I_2	**3**	0	0	0	**2**	0	**1**	0	**1**	0	0	0	0	0	**1**	0
A_3	30	31	32	33	34	35	36	37	38	39	3A	3B	3C	3D	3E	3F
I_3	0	0	0	0	**1**	0	0	0	**1**	0	0	**1**	0	0	0	0
A_4	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F
I_4	**1**	0	0	0	**1**	0	0	0	0	0	**1**	0	0	0	0	0
A_5	50	51	52	53	54	55	56	57	58	59	5A	5B	5C	5D	5E	5F
I_5	0	**1**	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0
A_6	60	61	62	63	64	65	66	67	68	69	6A	6B	6C	6D	6E	6F
I_6	**1**	**6**	0	**1**	0	**5**	0	**3**	**1**	**2**	0	**4**	**1**	**1**	0	**5**
A_7	70	71	72	73	74	75	76	77	78	79	7A	7B	7C	7D	7E	7F
I_7	**2**	**1**	**1**	**3**	**4**	**2**	0	**3**	0	**1**	**2**	**2**	**1**	**2**	**1**	0
A_8	80	81	82	83	84	85	86	87	88	89	8A	8B	8C	8D	8E	8F
I_8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A_9	90	91	92	93	94	95	96	97	98	99	9A	9B	9C	9D	9E	9F
I_9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A_{10}	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	AA	AB	AC	AD	AE	AF
I_{10}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A_{11}	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	BA	BB	BC	BD	BE	BF
I_{11}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A_{12}	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	CA	CB	CC	CD	CE	CF
I_{12}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A_{13}	D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	DA	DB	DC	DD	DE	DF
I_{13}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A_{14}	E0	E1	E2	E3	E4	E5	E6	E7	E8	E9	EA	EB	EC	ED	EE	EF

I_{14}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A_{15}	F0	F1	F2	F3	F4	F5	F6	F7	F8	F9	FA	FB	FC	FD	FE	FF
I_{15}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

W wypełnionej tabeli zaznaczone pogrubieniem i na czerwono są wszystkie elementy niezerowe. Szukamy maksimum – jest w wierszu I_6 i zostało już zaznaczone na zielono. Odpowiadający maksimum kod ASCII to 61, co oznacza literę **‘a’**.

W ten sposób wyznaczona została pierwsza litera klucza. Należy analogicznie kontynuować wyznaczanie kolejnych znaków klucza, pamiętając tylko o wyczyszczeniu używanej tablicy (wyzerowaniu kolumny Ilość).

Zadania.

1. Dokonać implementacji programu łamiącego XOR. Założyć, że program ma operować tylko na kryptogramach tekstów jawnych będących tekstami w języku polskim lub angielskim. Program powinien obsługiwać pliki (o dowolnej wielkości) i dokonywać operacji deszyfrowania po znalezieniu klucza. Klucz po jego znalezieniu powinien być wyświetlony na ekranie.
2. Znaleźć klucz oraz podać tekst jawny dla opisywanego w przykładzie kryptogramu.