Потокові шифри вразливі до атак, якщо один і той же ключ використовується два або більше разів.

Припустимо, ми надсилаємо повідомлення *A* і *B* однакової довжини, обидва зашифровані за допомогою одного ключа *K*. Нехай потоковий шифр видає рядок бітів *C (K)* такої ж довжини, що і повідомлення. Тоді зашифрованими версіями повідомлень є:

E (A) = A ***xor*** *C*

E (B) = B ***xor*** *C,*

де ***xor*** виконується побітно.

Операція ***xor*** є комутативною і має властивість *X* ***xor*** *X = 0*, звідси:

E (A) ***xor*** *E (B) = (A* ***xor*** *C)* ***xor*** *(B* ***xor*** *C) = A* ***xor*** *B* ***xor*** *C* ***xor*** *C = A* ***xor*** *B*

Іншими словами, якщо хтось перехоплює два повідомлення, зашифровані одним і тим же ключем, він може відновити *A* ***xor*** *B*.

Для таких випадків існує метод, який називається **crib dragging**, що може розкрити звичайний текст двох повідомлень, зашифрованих одним і тим же ключем, навіть не знаючи ключа.

Алгоритм розшифровки повідомлення за допомогою **crib dragging** приблизно наступний:

1. Підбирається слово, яке може з'явитися в одному з повідомлень – «*crib-слово*».
2. Зі слова з кроку 1 отримується масив байт (також можлива реалізація з hex рядками).
3. Виконується операція ***xor*** над масивами байт, що були отримані з двох зашифрованих повідомлень (*E (A)* ***xor*** *E (B)*).
4. Виконується операція ***xor*** над масивами з кроків 3 та 2.
5. Масив байт, отриманий на кроці 4, переводиться у рядок.
6. Якщо результат з кроку 5 є читабельним текстом, ми вгадуємо англійське слово та розширюємо наш пошук, переходячи до кроку 1 з новим отриманим *«crib-словом»*.
7. Якщо результат не є читабельним текстом, ми пробуємо виконати крок 4, зсуваючись по масиву з кроку 3 на наступну позицію.
8. Якщо ми пройшлися по масиву з кроку 3 до кінця і так і не натрапили на читабельний текст, переходимо на крок 1 і підбираємо нове «*crib-слово*».

Існують готові реалізації **crib dragging** калькуляторів, які були використані для виконання даної лабораторної роботи, наприклад:

- https://toolbox.lotusfa.com/crib_drag/
- https://lzutao.github.io/cribdrag/

Якщо вихідні зашифровані повідомлення різної довжини, довше повідомлення «обрізається» до розміру коротшого. Так атака виявить лише ту частину довшого повідомлення, яку покриває коротше. Але це не завадить розшифрувати «покриту» частину тим же методом.

Приклад:

Нехай маємо 2 повідомлення зашифровані одним і тим самим ключем без використання солі.

```
byte[] message1 = "Reusing the same key in streaming chiphers is a big mistake!".ToByteArray();
byte[] message2 = "Really? Well, I won't do that again".ToByteArray();
byte[] key = GenerateKey();

using var salsa20 = new Salsa20();

ICryptoTransform encryptor1 = salsa20.CreateEncryptor(key, new byte[8]);
ICryptoTransform encryptor2 = salsa20.CreateEncryptor(key, new byte[8]);

byte[] message1Enc = encryptor1.TransformFinalBlock(message1, 0, message1.Length);
byte[] message2Enc = encryptor2.TransformFinalBlock(message2, 0, message2.Length);
```

Виконуємо операцію ***xor*** над *message1Enc* і *message2Enc* .

```
byte[] mes1mes2 = Xor(message1Enc, message2Enc);
```

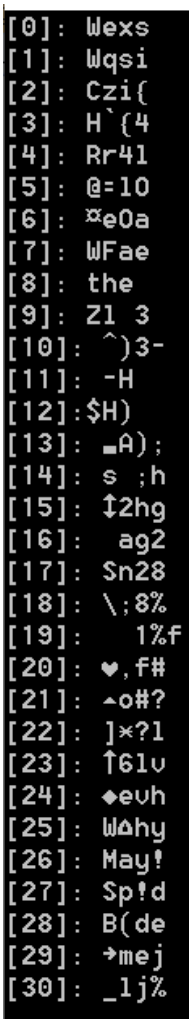
Беремо найбільш поширене в англійській мові слово *"the"* . Воно буде нашим першим «*crib-словом*». Виконуємо ***xor*** над *"the"* і *mes1mes2* послідовно просуваючись по *mes1mes2* на 1 символ та дивимося результат. У квадратних дужках зазначена величина – кількість символів зсуву.

```
string cribWord = "the";
for (int i = 0; i < mes1mes2.Length - cribWord.Length; i++)
{
    Console.WriteLine(
        $"[{i}]: {Encoding.UTF8.GetString(Xor(cribWord.ToByteArray(), mes1mes2.Skip(i).Take(mes1mes2.Length - i).ToArray()))}");
}
```

Отримуємо результат.

```
[0]: thq
[1]: t|z
[2]: `w`
[3]: kmr
[4]: qΔ=
[5]: c0e
[6]: ,hF
[7]: tKh
[8]: Wel
[9]: ya)
[10]: }$:
[11]: 87$
[12]: +)A
[13]: 5L
[14]: P-2
[15]: 1?a
[16]: #ln
[17]: pc;
[18]: Δ61
[19]: ×<,
[20]:  !o
[21]: =b×
[22]: ~'6
[23]: ;;e
[24]: 'hΔ
[25]: tra
[26]: nlp
[27]: p}(
[28]: a%m
[29]: 9`1
[30]: |ac
[31]: }n,
```

Бачимо, що на зсуві у 8 символів маємо буквосполучення *“Wel”*, що ймовірно може виявитися англійським словом *“Well”* . Пробуємо.



Наступні «*crib-слова*» обираються по такому ж принципу природнім підбором. Для таких цілей зручніше використовувати готові калькулятори, посилання на які вказані вище.

Зауваження. Функція *Xor* виглядає наступним чином.

```
4 references
public static byte[] Xor(byte[] first, byte[] second)
{
    int length = first.Length < second.Length ? first.Length : second.Length;

    var result = new byte[length];
    for(int i = 0; i < length; i++)
    {
        result[i] = (byte)(first[i] ^ second[i]);
    }

    return result;
}
```