Лабораторная работа № 8

Элементы криптографии. Шифрование (кодирование) различных исходных текстов одним ключом

Абакумов Егор Александрович

Содержание

List of Figures

List of Tables

Цель работы

Освоить на практике применение режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом.

Теоретическое описание

Простейшей и в то же время наиболее надёжной из всех схем шифрования является так называемая схема однократного использования, изобретение, которое чаще всего связывают с именем Г.С. Вернама.

Гаммирование – это наложение (снятие) на открытые (зашифрованные) данные криптографической гаммы, т.е. последовательности элементов данных, вырабатываемых с помощью некоторого криптографического алгоритма, для получения зашифрованных (открытых) данных. С точки зрения теории криптоанализа, метод шифрования случайной однократной равновероятной гаммой той же длины, что и открытый текст, является невскрываемым. Кроме того, даже раскрыв часть сообщения, дешифровщик не сможет хоть сколько-нибудь поправить положение – информация о вскрытом участке гаммы не даёт информации об остальных её частях [1].

“Наложение” гаммы – не что иное, как выполнение операции сложения по модулю 2 (xor) её элементов с элементами открытого текста. Эта операция в в математике обозначается знаком $\oplus$.

Гаммирование является симметричным алгоритмом. Поскольку двойное прибавление одной и той же величины по модулю 2 восстанавливает исходное значение, шифрование и дешифрование выполняется одной и той же программой.

К. Шенноном было доказано, что если ключ является фрагментом истинно случайной двоичной последовательности с равномерным законом распределением, причём его длина равна длине исходного сообщения, и используется этот ключ только один раз, после чего уничтожается, то такой шифр является абсолютно стойким, даже

если криптоаналитик располагает неограниченным ресурсом времени и неограниченным набором вычислительных ресурсов. Действительно, противнику известно только зашифрованное сообщение , при этом все различные ключевые последовательности возможны и равновероятны, а значит, возможны и любые сообщения , т.е. криптоалгоритм не даёт никакой информации об открытом тексте [2].

Ход работы

1. В первую очередь оговоримся, что использовать будем среду Jupyter Notebook и язык программирования Питон. Для выполнения задания нам необходимо будет подключить библиотеки random и string. Пишем блок необходимых функций, которые и реализуют всю логику программы: функция generate_new_key принимает на вход длину требуемого ключа и возвращает случайную строку символов, что и будет являться ключом; функция hexadecimal_form возвращает шестнадцатиричный вид подаваемой на вход строки; функция gamming будет выполнять основную роль - она выполняет непосредственно однократное гаммирование подаваемой строки с помощью ключа, передаваемого вторым аргументом (иллюстр. 0.1).

```python
import random
import string

def generate_new_key(size=6, chars = string.ascii_letters + string.digits):
    return ''.join(random.choice(chars) for _ in range(size))
def hexadecimal_form(s):
    return ' '.join("{:02x}".format(ord(c)) for c in s)

def gamming(fst_text, sec_text):
    fst_text_ascii = [ord(i) for i in fst_text]
    sec_text_ascii = [ord(i) for i in sec_text]
    return ''.join(chr(s ^ k) for s, k in zip(fst_text_ascii, sec_text_ascii))
```

Figure 0.1: Блок функций для дальнейшего криптоанализа

2. Теперь пишем блок обработки данных. Тут вводим переменные P1 и P2 для исходных текстов, переменную key для используемого ключа, C1 и C2 для шифротекстов P1 и P2 соответственно, переменную crypto_sum для результата гаммирования двух шифротекстов между собой (иллюстр. 0.2).

```python
P1 = "НаВашисходящийот1204"
P2 = "ВСеверныйфилиалБанка"
print("Исходные тексты:\n", "P1: ", P1, "\nP2: ", P2, sep='')
key = generate_new_key(len(P1))
print("Ключ для кодирования обоих текстов:")
print(key)
print("В шестнадцатиричном виде:")
print(hexadecimal_form(key))

C1 = gamming(P1, key)
C2 = gamming(P2, key)

print("\nШифротекст C1 для открытого текста P1 и ключа key")
print(C1, "--- C1")
print("Шифротекст C2 для открытого текста P2 и ключа key")
print(C2, "--- C2")

crypto_sum = gamming(C1, C2)

print("\nПолучим первый текст путем гаммирования двух шифровок и второго текста:")
print("Вычисленный P1:", gamming(crypto_sum, P2))
print("Получим второй текст путем гаммирования двух шифровок и первого текста:")
print("Вычисленный P2:", gamming(crypto_sum, P1))
```

Figure 0.2: Блок обработки данных и вывода требуемых значений

3. После запуска программы видим два исходных текста и ключ в символьной и шестнадцатиричной формах. После нам выводится оба шифротекста, получаемых гаммированием исходных текстов P1 и P2 одним известным ключом key. Далее видим вычисляемые тексты P1 и P2 (иллюстр. 0.3). Вычисляются они с помощью суммы по модулю 2 обеих шифровок и одного из исходных текстов в соответствии с формулой: $C_1 \oplus C_2 \oplus P_1 = P_1 \oplus P_2 \oplus P_1 = P_2$. Таким образом, зная шифротексты и хотя бы часть одного из исходных текстов, можно расшифровать оба исходных текста.

```
Исходные тексты:
P1: НаВашисходящийот1204
P2: ВСеверныйфилиалБанка
Ключ для кодирования обоих текстов:
PUr989yaBY3TasPykjM6
В шестнадцатиричном виде:
50 55 72 39 38 39 79 61 42 59 33 54 61 73 50 79 6b 6a 4d 36

Шифротекст C1 для открытого текста P1 и ключа key
эҝѠЉΨЁиФѺѫѺНљъѮлZX}⍰ --- C1
Шифротекст C2 для открытого текста P2 и ключа key
тVчЋЙѹфЪѻНЋѮљуӿѬħїѶI --- C2

Получим первый текст путем гаммирования двух шифровок и второго текста:
Вычисленный P1: НаВашисходящийот1204
Получим второй текст путем гаммирования двух шифровок и первого текста:
Вычисленный P2: ВСеверныйфилиалБанка
```

Figure 0.3: Вывод значений и результат работы

Выводы

В ходе работы мы успешно на практике освоили применение режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом.

Ответы на контрольные вопросы

1. Как, зная один из текстов (P1 или P2), определить другой, не зная при этом ключа?

- Для этого необходимо прогаммировать один шифротекст вторым, а после прогаммировать результат одним из исходных текстов. Таким образом мы получим другой исходный текст.

2. Что будет при повторном использовании ключа при шифровании текста?

- Если речь о разных текстах, то мы создадим пару взаимосвязанных текстов, которые будут подвержены риску взлома при компроментации одного из исходных текстов. Если же речь об одном тексте, то мы из шифрованного текста обратно получим исходный нешифрованный.

3. Как реализуется режим шифрования однократного гаммирования одним ключом двух открытых текстов?

- Поочередно гаммируем каждый текст одним ключом.

4. Перечислите недостатки шифрования одним ключом двух открытых текстов.

- Подверженность взлому, шифр становится абсолютно вскрываемым. При утечке же хотя бы части одного из исходных текстов злоумышленник сможет расшифровать все тексты.

5. Перечислите преимущества шифрования одним ключом двух открытых текстов.

- Можно сократить издержки по доставке ключей сторонам, либо вообще исключить их, если ключ использовать все время.

Список литературы

1. Шнайер, Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си / Б. Шнайер. – М. : Триумф, 2002. – 816 с.

2. Харин, Ю.С. Математические и компьютерные основы криптологии : учебное пособие / Ю.С. Харин, В.И. Берник, Г.В. Матвеев, С.В. Агиевич. – Мн. : Новое знание, 2003. – 382 с.

3. Д. С. Кулябов, А. В. Королькова, М. Н. Геворкян. Информационная безопасность компьютерных сетей: лабораторные работы. // Факультет физико-математических и естественных наук. М.: РУДН, 2015. 64 с..