0.1 Front matter

title: “Лабораторная работа №7” subtitle: “Основы информационной безопасности” author: “Пинега Белла Александровна”

0.2 Generic otions

lang: ru-RU toc-title: “Содержание”

0.3 Bibliography

bibliography: bib/cite.bib csl: pandoc/csl/gost-r-7-0-5-2008-numeric.csl

0.4 Pdf output format

toc: true # Table of contents toc-depth: 2 lof: true # List of figures lot: true # List of tables fontsize: 12pt linestretch: 1.5 papersize: a4 documentclass: scrreprt ## I18n polyglossia polyglossia-lang: name: russian options: - spelling=modern - babelshorthands=true polyglossia-otherlangs: name: english ## I18n babel babel-lang: russian babel-otherlangs: english ## Fonts mainfont: PT Serif romanfont: PT Serif sansfont: PT Sans monofont: PT Mono mainfontoptions: Ligatures=TeX romanfontoptions: Ligatures=TeX sansfontoptions: Ligatures=TeX,Scale=MatchLowercase monofontoptions: Scale=MatchLowercase,Scale=0.9 ## Biblatex biblatex: true biblio-style: “gost-numeric” biblatexoptions: - parentracker=true - backend=biber - hyperref=auto - language=auto - autolang=other* - citestyle=gost-numeric ## Pandoc-crossref LaTeX customization figureTitle: “Рис.” tableTitle: “Таблица” listingTitle: “Листинг” lofTitle: “Список иллюстраций” lotTitle: “Список таблиц” lolTitle: “Листинги” ## Misc options indent: true header-includes: -

keep figures where there are in the text

- # keep figures where there are in the text

1 Цель работы

Освоить на практике применение режима однократного гаммирования

2 Задание

Нужно подобрать ключ, чтобы получить сообщение «С Новым Годом, друзья!». Требуется разработать приложение, позволяющее шифровать и дешифровать данные в режиме однократного гаммирования. Приложение должно: 1. Определить вид шифротекста при известном ключе и известном откры- том тексте. 2. Определить ключ, с помощью которого шифротекст может быть преоб- разован в

некоторый фрагмент текста, представляющий собой один из возможных вариантов прочтения открытого текста

3 Теоретическое введение

Предложенная Г. С. Вернамом так называемая «схема однократного ис- пользования (гаммирования)» (рис. 7.1) является простой, но надёжной схе- мой шифрования данных. Гаммирование представляет собой наложение (снятие) на открытые (за- шифрованные) данные последовательности элементов других данных, по- лученной с помощью некоторого криптографического алгоритма, для по- лучения зашифрованных (открытых) данных. Иными словами, наложение гаммы — это сложение её элементов с элементами открытого (закрытого) текста по некоторому фиксированному модулю, значение которого пред- ставляет собой известную часть алгоритма шифрования. В соответствии с теорией криптоанализа, если в методе шифрования ис- пользуется однократная вероятностная гамма (однократное гаммирование) той же длины, что и подлежащий сокрытию текст, то текст нельзя раскрыть. Даже при раскрытии части последовательности гаммы нельзя получить ин- формацию о всём скрываемом тексте. Наложение гаммы по сути представляет собой выполнение операции сложения по модулю 2 (XOR) (обозначаемая знаком $\oplus$) между элементами гаммы и элементами подлежащего сокрытию текста. Напомним, как рабо- тает операция XOR над битами: $0 \oplus 0 = 0$, $0 \oplus 1 = 1$, $1 \oplus 0 = 1$, $1 \oplus 1 = 0$. Такой метод шифрования является симметричным, так как двойное при- бавление одной и той же величины по модулю 2 восстанавливает исходное Рис. 7.1. Схема однократного использования Вернама 1При составлении работы использовалось пособие [1]. 46 Кулябов Д. С., Королькова А. В., Геворкян М. Н. значение, а шифрование и расшифрование выполняется одной и той же про- граммой. Если известны ключ и открытый текст, то задача нахождения шифротек- ста заключается в применении к каждому символу открытого текста следу- ющего правила: $Ci = Pi \oplus Ki$, (7.1) где Ci — i-й символ получившегося зашифрованного послания, Pi — i-й символ открытого текста, Ki — i-й символ ключа, $i = 1, m$. Размерности открытого текста и ключа должны совпадать, и полученный шифротекст будет такой же длины. Если известны шифротекст и открытый текст, то задача нахождения ключа решается также в соответствии с (7.1), а именно, обе части равен- ства необходимо сложить по модулю 2 с Pi: $Ci \oplus Pi = Pi \oplus Ki \oplus Pi = Ki$, $Ki = Ci \oplus Pi$. Открытый текст имеет символьный вид, а ключ — шестнадцатеричное представление. Ключ также можно представить в символьном виде, вос- пользовавшись таблицей ASCII-кодов. К. Шеннон доказал абсолютную стойкость шифра в случае, когда одно- кратно используемый ключ, длиной, равной длине исходного сообщения, является фрагментом истинно случайной двоичной последовательности с равномерным законом распределения. Криптоалгоритм не даёт никакой ин- формации об открытом тексте: при известном зашифрованном сообщении C все различные ключевые последовательности K возможны и равноверо- ятны, а значит, возможны и любые сообщения P . Необходимые и достаточные условия абсолютной стойкости шифра: – полная случайность ключа; – равенство длин ключа и открытого текста; – однократное использование ключа. Рассмотрим пример. Ключ Центра: 05 0C 17 7F

0E 4E 37 D2 94 10 09 2E 22 57 FF C8 0B B2 70 54 Сообщение Центра: Штирлиц – Вы Герой!! D8 F2 E8 F0 EB E8 F6 20 2D 20 C2 FB 20 C3 E5 F0 EE E9 21 21 Зашифрованный текст, находящийся у Мюллера: DD FE FF 8F E5 A6 C1 F2 B9 30 CB D5 02 94 1A 38 E5 5B 51 75 Дешифровальщики попробовали ключ: 05 0C 17 7F 0E 4E 37 D2 94 10 09 2E 22 55 F4 D3 07 BB BC 54 и получили текст: D8 F2 E8 F0 EB E8 F6 20 2D 20 C2 FB 20 C1 EE EB E2 E0 ED 21 Штирлиц - Вы Болван!

4 Выполнение лабораторной работы

1. Код программы:

```python
import random
import string

def generate_key(size) :
    characters = string.ascii_letters + string.digits
    return ''.join(random.choice (characters) for _ in range (size))

def text_to_binary(text):
    return ''.join(format(ord (char), '08b' )for char in text)

def binary_text (binare_str):
    binary_chunks = [binary_str[i:i+8] for i in range(0, len(binary_str),8)]
    return ''.join(chr(int (chunk, 2))for chunk in binary_chunks)

def xor_encrypt (text, key) :
    encrypted = [ord(a) ^ ord (b) for a, b in zip(text, key)]
    return ''.join(chr(encrypted_char)for encrypted_char in encrypted)

msg = "С Новым годом, друзья!"

key = generate_key(len (msg))

print ("Ключ:", key)

msg2 = xor_encrypt (msg, key)

binary = text_to_binary (msg2)

print ("Зашифрованный текст:", binary)

msg3 = xor_encrypt (msg2, key)

print ("Расшифрованный текст:", msg3)
```

2. Результат, ключ, с помощью которого шифротекст может быть преобразован в некоторый фрагмент текста, представляющий собой один из возможных вариантов прочтения открытого текста

5 Выводы

Я освоила на практике применение режима однократного гаммирования

Список литературы