Университет ИТМО

Факультет программной инженерии и компьютерной техники

Информационная безопасность

Отчет по лабораторной работе № 3

«Поточное симметричное шифрование»

Вариант 5

Выполнил студент группы P34302

Ким Даниил Кванхенович

Проверил преподаватель

Рыбаков Степан Дмитриевич

Санкт-Петербург 2024

Содержание

Цель работы:

Изучение структуры и основных принципов работы современных алгоритмов поточного симметричного шифрования, приобретение навыков программной реализации поточных симметричных шифров.

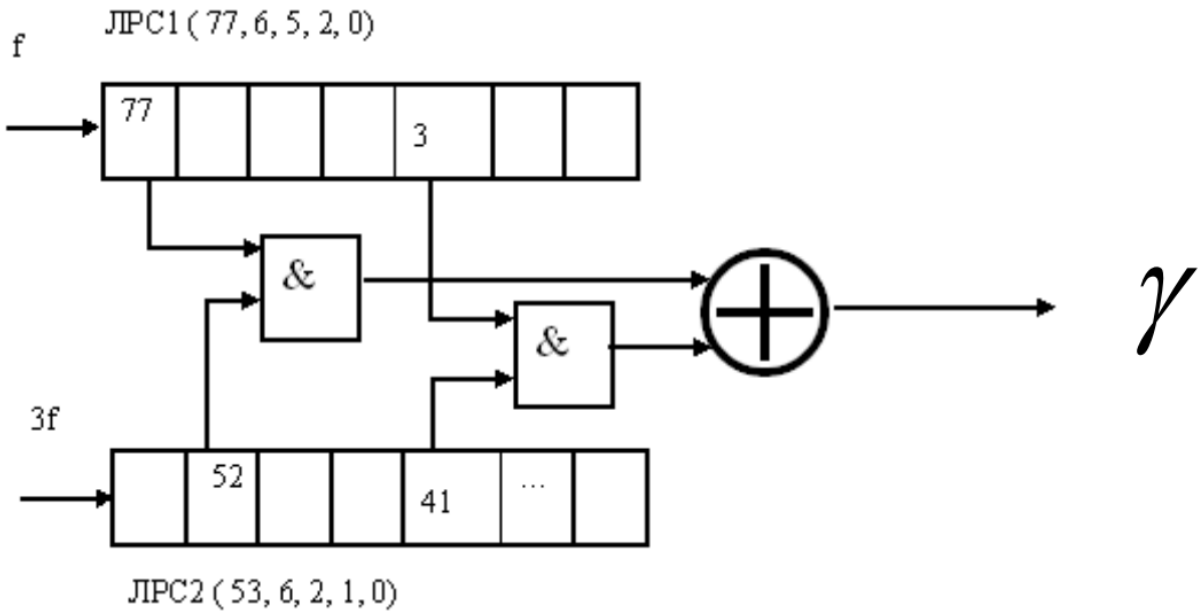
Порядок выполнения работы:

1. Ознакомьтесь с теоретическими основами шифрования данных.
2. Получите вариант задания у преподавателя.
3. Напишите программу согласно варианту задания.
4. Отладьте разработанную программу и покажите результаты работы программы преподавателю.
5. Составьте отчет по лабораторной работе.

Данные варианта:

Вариант 5.

Реализовать в программе поточное кодирование текста, вводимого с клавиатуры, с помощью заданной нелинейной схемы, использующей разные частоты тактирования ЛРС.



Выполнение:

Листинг разработанного модуля LFSR:

Разработанный модуль представляет из себя регистр сдвига, предоставляющий гаммирующий ключевой поток для потокового шифрования. В общем, регистр сдвига можно охарактеризовать его внутренним состоянием и набором отводов.

```
public class LFSR {
    private final LinkedList<Integer> state;
    private final TreeSet<Integer> feedbacks;

    public LFSR(Collection<Integer> initState,
                Collection<Integer> feedbacks,
                boolean feedbacksHighToLow,
                boolean feedbacksStartZero) {…}


    public int shiftLeft(boolean verbose) {…}

    public int shiftRight(boolean verbose) {…}

    public int get(int index,
        boolean indexHighToLow, boolean indexStartZero){…}

    public String getFeedbackString () {…}

    @Override
    public String toString() {…}
}
```

Для генерации шифрующего потока используется один из двух основных методов:

- Сдвиг влево ;
- Сдвиг вправо ;

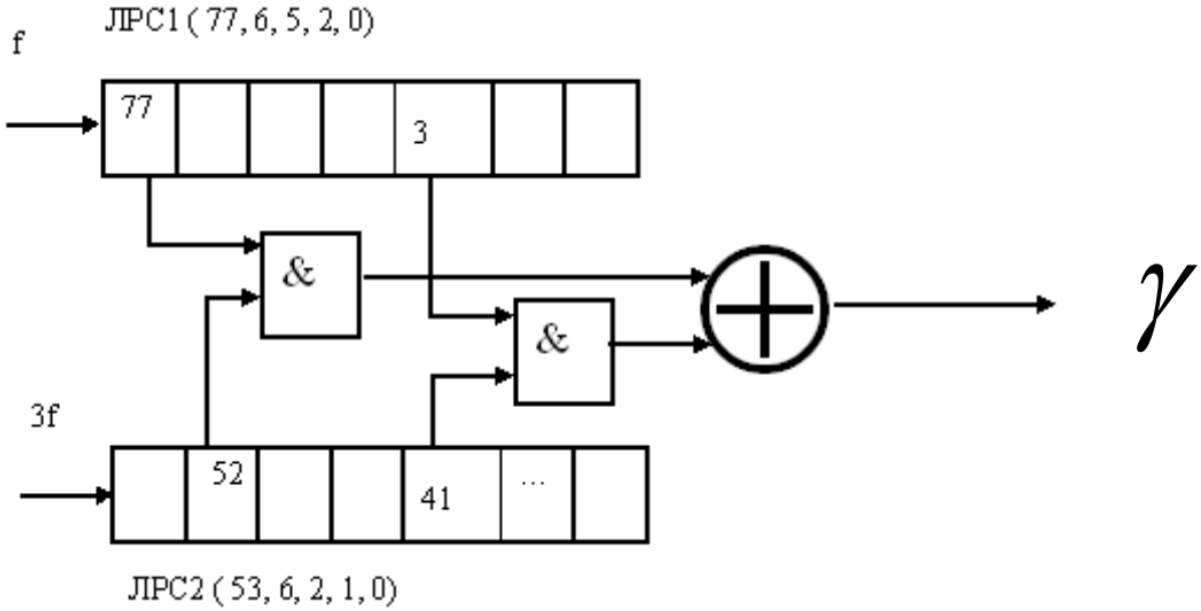
При этом выходу регистра соответствует крайний левый, либо крайний правый бит соответственно. Остальные ячейки двигаются по направлению сдвига, а значение, поступающее на вход регистра, формируется из его отводов.

В качестве нелинейной функции обратной связи, формирующей значение входа, используется исключающая дизъюнкция (xor).

```
public int shiftRight(boolean verbose) {
    int generatedValue = 0;
    int outputValue = state.getLast();

    for (int feedback: feedbacks) {
        generatedValue ^= state.get(feedback);
    }

    state.addFirst(generatedValue);
    state.pollLast();

    if (verbose) System.out.printf("%s -> %d\n", this,
outputValue);

    return  outputValue;
}
```

Описание алгоритма работы комбинации РСОС:

Для обеспечения большей криптостойкости сдвиговых регистров с обратной связью их следует комбинировать. В данной лабораторной работе было предложено реализовать комбинацию двух регистров со своей частотой тактирования и своими наборами отводов.



Таким образом, первый регистр имеет разрядность в 77 ячеек и 4 отвода: ячейки 77, 6, 5, 3. Ячейка 0 является его выходом. Второй регистр имеет 53 ячейки, 4 отвода: 53, 6, 2, 1 и выход.

Для генерации ключевого потока используется комбинация из 77 и 3 ячейки первого регистра и 52 и 41 ячейки второго регистра соединенные нелинейными функциями конъюнкции (and) и исключающей дизъюнкцией (xor).

Вывод программы:

Демонстрация работы сдвигового регистра:

1. Регистр заполняется начальным состоянием и индексами, соответствующими номерами ячеек, служащих в качестве отводов.
2. В качестве направления используется сдвиг в право.
3. После 4 сдвига видно, что состояние регистра соответствует исходному. Период данного регистра равен 4.

```
LinkedList<Integer> initState = new LinkedList<>();
initState.add(0);
initState.add(0);
initState.add(1);
initState.add(1);


LinkedList<Integer> feedbacks = new LinkedList<>();
feedbacks.add(4);
feedbacks.add(3);
feedbacks.add(2);

LFSR register = new LFSR(initState, feedbacks, true, false);


System.out.println(register.getFeedbackString());
System.out.println(register);

for (int i = 0; i < 5; i++) {
    register.shiftRight(true);
}
```

```
_ _ _
0 0 1 1
1 0 0 1 -> 1
1 1 0 0 -> 1
0 1 1 0 -> 0
0 0 1 1 -> 0
1 0 0 1 -> 1
```

Демонстрация работы комбинации сдвиговых регистров:

```
LinkedList<Integer> initState1 = new LinkedList<>();
for (int i = 0; i < 77; i++) {
    initState1.add(i);
}
LinkedList<Integer> initState2 = new LinkedList<>();
for (int i = 0; i < 53; i++) {
    initState2.add(i);
}

LinkedList<Integer> feedbacks1 = new LinkedList<>(Arrays.asList(77, 6,
5, 2));
LinkedList<Integer> feedbacks2 = new LinkedList<>(Arrays.asList(53, 6,
2, 1));

LFSR register1 = new LFSR(initState1, feedbacks1, true, false);
System.out.println(register1.getFeedbackString());
System.out.println(register1);
LFSR register2 = new LFSR(initState2, feedbacks2, true, false);
System.out.println(register2.getFeedbackString());
System.out.println(register2);

for (int i = 0; i < 1000; i++) {
    register1.shiftRight(false);
    int out11 = register1.get(77, true, false);
    int out12 = register1.get(3, true, false);

    register2.shiftRight(false);
    int out21 = register2.get(52, true, false);
    int out22 = register2.get(41, true, false);

    int and1 = out11 & out21;
    int and2 = out12 & out22;

    int xor = and1 ^ and2;
    System.out.print(xor+", ");
}
```

Сгенерированная последовательность имеет следующий вид:

```
9, 0, 4, 8, 1, 4, 8, 0, 17, 48, 20, 48, 36, 52, 40, 48, 48, 16,
52, 24, 52, 36, 56, 32, 48, 48, 36, 32, 4, 36, 8, 32, 32, 0, 36,
8, 36, 4, 40, 0, 16, 48, 36, 16, 4, 20, 8, 16, 16, 7, 16, 0, 20,
30, 25, 16, 0, 20, 1, 0, 1, 10, 11, 2, 8, 4, 5, 0, 0, 10, 11, 2,
0, 4, 12, 8, 4, 0, 10, 2, 28, 32, 4, 0, 36, 40, 26, 50, 12, 8 …
```

Вывод:

В ходе данной лабораторной работы был закреплен материал об алгоритмах симметричного поточного шифрования. Был получен опыт их программной реализации. Была проиллюстрирована работа как единственного сдвигового регистра с обратной связью с малым периодом генерируемой последовательности, так их комбинации.