МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ
ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
ГРАЖДАНСКОЙ АВИАЦИИ

Э.А. Болелов,
Е.Б. Биктеева

ОСНОВЫ ЗАЩИТЫ ИНФОРМАЦИИ НА ТРАНСПОРТЕ

Учебно-методическое пособие
по выполнению лабораторных работ

для студентов V курса
специальности 25.05.03
всех форм обучения

Москва
2019

ФЕДЕРАЛЬНОЕ АГЕНТСТВО ВОЗДУШНОГО ТРАНСПОРТА

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ГРАЖДАНСКОЙ АВИАЦИИ (МГТУ ГА)»

Кафедра технической эксплуатации радиоэлектронного оборудования воздушного транспорта

Э.А. Болелов, Е.Б. Биктеева

ОСНОВЫ ЗАЩИТЫ ИНФОРМАЦИИ НА ТРАНСПОРТЕ

Учебно-методическое пособие
по выполнению лабораторных работ

для студентов
специальности 25.05.03
всех форм обучения

Москва
2019

ББК 6Ф7.3
Б-79

Рецензент:
Полосин С.А. – ведущий специалист ГосНИИ АС

Болелов Э.А.
Б-79 Основы защиты информации на транспорте: учебно-методическое пособие по выполнению лабораторных работ./ Э.А. Болелов, Е.Б. Биктеева. – Воронеж: ООО «МИР», 2019. – 48 с.

Данное учебно-методическое пособие издается в соответствии с рабочей программой учебной дисциплины «Моделирование систем и процессов в задачах эксплуатации транспортного радиооборудования» по учебному плану для студентов V курса специальности 25.05.03 всех форм обучения.

Рассмотрено и одобрено на заседании кафедры 19.06.2019 г. и методического совета 19.06.2019 г.

В авторской редакции.

Подписано в печать 07.10.2019 г.
Формат 60x84/16 Печ.л. 3 Усл. печ. л. 2,79
Заказ 525/2072 Тираж 80 экз.

Московский государственный технический университет ГА
125993 Москва, Кронштадтский бульвар, д.20

Отпечатано ООО «МИР»
394033, г. Воронеж, Ленинский пр-т 119А, лит. Я, оф. 215
Тел.: 8 (958) 649-53-31 Email: 89586495331@mail.ru

Содержание

Лабораторная работа №1.

Исследование стойкости симметричных криптосистем

Цель работы - изучение классических криптографических алгоритмов моноалфавитной подстановки, многоалфавитной подстановки и перестановки для защиты текстовой информации. Использование гистограмм, отображающих частоту встречаемости символов в текст для криптоанализа классических шифров.

Изучение методов шифрования/расшифрования перестановкой символов, подстановкой, гаммированием, использованием
таблицы Виженера. Исследование и сравнение стойкости различных методов на основе атак путем перебора всех возможных ключей.

Описание лабораторной работы

1. Для выполнения лабораторной работы необходимо запустить программу L_LUX.EXE. На экране дисплея появляется окно с размещенным в его центре текстовым редактором (для отображения зашифрованных и расшифрованных текстов), в верхней строке окна расположено главное меню, позволяющее пользователю выполнить требуемое действие. Чуть ниже основного меню размещена панель инструментов (для управления быстрыми командными кнопками и другими «горячими» элементами управления), а в самом низу окна, под текстовым редактором, находится строка состояния, в которой указывается подсказка и выводится дополнительная информация. Клавиши панели инструментов для удобства снабжены всплывающими подсказками.

Для того чтобы попасть в основное меню, необходимо нажать клавишу F10. Передвижение по главному меню осуществляется клавишами перемещения курсора. Чтобы вызвать пункт меню, нужно нажать клавишу ENTER, вернуться в главное меню или вовсе выйти из него — ESC.

Рассмотрим более подробно каждый из пунктов главного меню.

Редактор.

Данный пункт основного меню содержит подпункты: создать документ, открыть файл, сохранить файл, выход из программы.

Предварительно, сразу после запуска программы, текстовый редактор недоступен, также недоступными являются почти все пункты главного меню, кроме создания документа. открытия файла, выхода из программы, информации о программе, и большая часть клавиш панели управления, за исключением создания документа, открытия файла и выхода из программы.

Создать документ (Ctrl+N) — при выборе данного подпункта становится доступна работа с тестовым редактором (пользователь получает право создать свой текстовый файл, который впоследствии можно будет использовать при работе с программой), также появляется возможность использовать все недоступные до этого пункты основного меню и клавиши панели управления.

Открыть файл (Ctrl+L) — при выборе этого пункта появляется диалоговое окно, предоставляющее возможность выбора файла для загрузки. При этом содержимое файла будет отображено в окне редактора текстов.

Аналогично пункту СОЗДАТЬ ДОКУМЕНТ доступным для работы становится текстовый редактор с отображаемым текстом, а также появляется возможность использовать все недоступные до этого пункты
основного меню и клавиши панели управления.

Сохранить файл (Ctrl+S) — при выборе этого пункта появляется диалоговое окно, позволяющее сохранить на диске содержимое редактора текстов.

Выход из программы (Ctrl+X) — при выборе этого пункта появляется диалоговое окно, позволяющее выйти из программы.

Гистогрармма. Вывод на экран двух гистограмм, отображающих частоту встречаемости символов в тексте.

ВНИМАНИЕ! До выполнения шифрования и дешифрования вызывать гистограмму не имеет смысла, так как еще не сформированы тексты, для которых будет просматриваться гистограмма.

Имеется возможность просмотра следующих сочетаний гистограмм:

- исходного и зашифрованного файла;
- зашифрованного и расшифрованного файла;
- стандартного распределения и зашифрованного текста;
- стандартного распределения и расшифрованного текста.

С целью масштабирования в гистограммах используются левая и правая клавиши мыши. Например, после шифрования текста большого объема пользователь хочет посмотреть гистограммы исходного и зашифрованного

файла. Поскольку размеры текста достаточно большие, то на экран будут выведены две гистограммы с большим количеством столбцов в каждой (столбец соответствует одному символу текста), однако трудно будет сказать, какой из этих столбцов соответствует тому или иному символу текста и какова частота встречаемости данного символа. Поэтому у пользователя есть возможность увеличить масштаб любой из двух гистограмм для более точного определения требуемого значения частоты встречаемости конкретного символа. Для этого необходимо навести указатель мыши на левую границу того участка гистограммы, который требуется увеличить, затем нажать левую клавишу мыши и, не отпуская ее, растянуть прямоугольник так, чтобы его нижний правый угол совпал с правой границей увеличиваемого участка гистограммы. После этого следует отпустить левую клавишу мыши, и на экране появится увеличенное изображение нужного участка. Нажав и не отпуская правую клавишу мыши, можно перемещать гистограмму в любом направлении с целью изучения всего полученного распределения в увеличенном масштабе.

Для того чтобы от увеличенного масштаба вернуться к исходному виду, нужно навести указатель мыши на гистограмму, затем нажать
левую клавишу мыши и, не отпуская ее, снизу вверх растянуть небольшой по размерам прямоугольник, после этого следует отпустить левую клавишу мыши, и на экране появится исходное изображение гистограммы.

Шифрование.

Выполнение шифрования текстового файла осуществляется одним из семи методов, рассматриваемых в лабораторной
работе:

1) одноалфавитный (с фиксированным смещением);
2) одноалфавитный с задаваемым смешением (от 2 до 20);
3) перестановка символов;
4) по дополнению до 255 (инверсный);
5) многоалфавитный (с фиксированным ключом);
6) многоалфавитный с ключом фиксированной длины;
7) многоалфавитный с ключом произвольной длины.

Выбор метода шифрования производится как мышкой, так и клавишами перемещения курсора и клавишей ENTER. Затем появляется
окно, в котором в зависимости от метода шифрования требуется указать те или иные параметры и либо подтвердить процесс кодировки, либо отказаться от него. После этого в окне редактора будет выдан зашифрованный текст.

Расшифрование.

Аналогично предыдущему пункту выбирается метод расшифрования (должен соответствовать методу, которым был зашифрован файл). Снова появляется окно, в котором в зависимости от метода расшифрования требуется указать те или иные параметры и либо подтвердить процесс расшифрования, либо отказаться от него. После этого в окно редактора будет выдан расшифрованный текст. При правильном расшифровании полученный текст совпадает с исходным.

Дополнительная информация.

Программа предусматривает возможность посмотреть дополнительную информацию («Помощь Ctrl+F9»),
справочную информацию об используемых методах шифрования («О методах Ctrl+F10»), сведения о программе («О программе Ctrl+F11»).

Пример работы с программой

Рассмотрим одноалфавитное шифрование с фиксированным ключом.

Нажав клавиши Ctrl+L либо выбрав в меню пункт ОТКРЫТЬ ФАЙЛ, загрузите в окно редактора исходный текст.

Затем вызовите пункт меню ШИФРОВАНИЕ, выберите одноалфавитный метод (с фиксированным смещением). В появившемся окне нажмите клавишу ЗАШИФРОВАТЬ. После того как шифрование выполнено, можно в редакторе просмотреть зашифрованный текст.

Перейдите к пункту меню ГИСТОГРАММА. Выберите тип гистограмм, отображающий гистограммы исходного и шифрованного файлов. Проанализируйте гистограммы. Они должны иметь примерно
одинаковый вид.

Чтобы узнать ключ шифра (смещение второго алфавита относительно первого), необходимо по гистограммам найти символы, имеющие одинаковую частоту встречаемости. Например, самый частый символ в первой гистограмме при шифровании должен перейти в самый частый символ во второй гистограмме. Таким образом, найдя два самых часто встречаемых символа в обеих гистограммах, можно по стандартной таблице ASCII-кодов вычислить смещение. Зная смещение и таблицу кодировки символов, текст можно легко расшифровать. Вызвав пункт меню ДЕШИФРОВАНИЕ, можно провести те же действия в автоматическом режиме.

Примечание. При шифровании и расшифровании из таблицы кодировки не используются символы с кодами 176—223 и 240—255, т.е. при ручной расшифровке зги символы следует пропускать и считать, что, например, символ «Я» имеет код не 159, а 223, аналогично «п» не 175, а 239.

Иногда в гистограммах под столбцами, показывающими частоту встречаемости символов, изображены не сами символы, а их табличные коды в квадратных скобках.

Ниже провидено описание «горячих» клавиш и их использование при выполнении различных действий:

Shift+F10 - о программе;

Ctrl+X - выход из программы:

Ctrl+N - new - ФАЙЛ/СОЗДАТЬ;

Ctrl+L - load - ФАЙЛ/ОТКРЫТЬ;

Ctrl+ S - save - ФАЙЛ/СОХРАНИТЬ.

Шифрование:

Ctrl+F1 - одноалфавитный метод (с фиксированным смещением);

Ctrl+F2 - одноалфавитный с задаваемым смешением (от 2 до 20);

Ctrl+F3 - перестановка символов;

Ctrl+F4 - по дополнению до 255 (инверсный метод);

Ctrl+F5 - многоалфавитный метод с фиксированным ключом;

Ctrl+F6 - многоалфавитный метод с ключом фиксированной длины;

Ctrl+F7 - многоалфавитный метод с ключом произвольной длины.

Расшифрование:

Shift+F1- одноалфавитный метод (с фиксированным смещением);

Shift+F2 - одноалфавитный с задаваемым смещением (от 2 до 20);

Shift+F3 - перестановка символов;

Shift+F4 - по дополнению до 255 (инверсный метод);

Shift+F5 - многоалфавитный метод с фиксированным ключом;

Shift+F6 - многоалфавитный метод с ключом фиксированной длины;

Shift+F7 - многоалфавитный метод с ключом произвольной длины.

Гистограммы:

Shift+Ctrl+F1 - исходного и шифрованного файла;

Shift+Ctrl+F2 - шифрованного и расшифрованного файла;

Shift+Ctrl+F3 - стандартного распределения и шифрованного текста;

Shift+Ctrl+F4 - стандартного распределения и расшифрованного текста.

Помощь:

Ctrl+F9 - помощь;
Ctrl+F10 – о методах;
Ctrl+F11 – о программе.

2. В лабораторной работе рассматривается способ вскрытия шифра, основанный на переборе всех вариантов ключа. Критерием правильности варианта служит наличие в тексте «вероятного слова». Перебирается множество всех возможных ключей, шифрованный текст расшифровывается на каждом ключе. В получившемся «псевдооткрытом» тексте ищется вероятное слово. Если такого слова нет, текущий текст бракуется и осуществляется переход к следующему ключу. Если такое слово найдено, на экран выводится вариант ключа. Затем перебор ключей продолжается до тех пор, пока не исчерпается все множество вариантов. Возможно обнаружение нескольких ключей, при которых в «псевдооткрытых текстах» имеется вероятное слово.

После завершения перебора необходимо расшифровать текст на найденных ключах. «Псевдооткрытый текст» выводится на экран для визуального контроля. Если оператор признает текст открытым, то работа по вскрытию заканчивается. Иначе данный вариант ключа бракуется и осуществляется переход к следующему ключу.

Для выполнения лабораторной работы необходимо запустить программу LAB_RAB.EXE, используемую для шифрования/расшифрования, а также дешифрования (методом протяжки вероятного слова) файлов.

Система реализует мелющие функции:

- ввод, удаление и селекция ключей пользователя;
- поддержка списка ключей;
- шифрование и расщифрование текста;
- дешифрование текста путем подбора ключей, методом протяжки вероятного слова.

Система поддерживает следующие методы криптографического преобразования информации:

- замена;
- перестановка;
- гаммирование;
- таблица Виженера.

При запуске утилит шифрования и расшифрования у пользователя

запрашивается подтверждение на правильность выбранного метода для работы и соответствия заданного ключа целям пользователя (также всегда при изменении файла в текстовом редакторе выдается запрос на сохранение изменений при каждом шаге, дальнейшее развитие которого приведет к необратимым изменениям в файле и потере изначальной информации).

Описание интерфейса*:*

- окно текстового редактора с широким набором дополнительных функций;
- таблица всех ключей, введенных в систему с быстрым доступом для ввода, удаления или выбора текущего ключа;
- список всех методов шифрования для быстрого и удобного переключения между ними;
- основное меню (вверху экрана);
- дополнительное меню (вызывается нажатием правой кнопки мыши);
- набор вспомогательных кнопок для быстрого и удобного интерфейса;
- поля вывода текущего состояния системы;

- текущий ключ,
- вероятное слово,
- сила ключа для протяжки.

Пример работы с программой

ВНИМАНИЕ! Будьте внимательны при установке параметров работы, так как в процессе вычисления по ходу работы эти параметры изменить уже не удастся. После запуска программы абсолютно все рабочие поля пустые и необходимо провести первоначальные настройки для работоспособности системы.

Алгоритм работы с программой:

1. вводится список ключей;
2. вводится вероятное слово (необязательно вначале, до его ввода все меню запуска протяжки все равно недоступны);
3. выбирается необходимый метод шифрования;
4. загружается исходный или зашифрованный файл (открываются соответствующие меню для шифрования и расшифрования);

5. запускается необходимый процесс;

- шифрование,
- расщифрование,
- протяжка вероятного слова,
- конвертация текста;

6. продолжение работы в любом порядке в соответствии с описанными пунктами:

7. при завершении работы не забудьте сохранить необходимые результаты (при закрытии и загрузке новых файлов система автоматически запрашивает подтверждение на запись).

Шифрование:

1. Открыть файл.
2. Внести необходимые изменения.
3. Настроить соответствующие параметры;

▪ тип шифрования;

▪ ключ, пр.

4. Запустить процесс шифрования через пункт меню УТИЛИТЫ/ ЗАШИФРОВАТЬ F5.

ВНИМАНИЕ! При шифровании файла все внесенные пользователем изменения до текущего момента времени будут сохранены на жестком диске.

Расшифрование:

1. Открыть файл.
2. Произвести необходимые изменения.
3. Настроить соответствующие параметры: тип шифрования. пр.
4. Запустить процесс расшифрования через пункт меню УТИЛИТЫ/РАСШИФРОВАТЬ F6.

ВНИМАНИЕ! При расшифровании файла все проведенные пользователем изменения до текущего момента времени будут сохранены на жестком диске.

Протяжка вероятного слова (дешифроваппе)

ВНИМАНИЕ! Мощность ключа задается заранее в опции «сила ключа».

Длина ключа значительно влияет на время протяжки вероятного слова (в худшем случае имеем дело с логарифмическим алгоритмом).

1. Вводится вероятное слово (длиной от 1 (3) до 9).
2. Для отделения вновь найденных ключей от предыдущих между ними добавляется надпись «подбор».

3. Перебираются ключи.

4. Расшифровывается вся первая строка текста по текущему ключу.

5. Порциями, равными длине вероятного слова, сравнивается содержимое этой строки со значением вероятного слова.

6. Если найдено хоть одно совпадение, запоминается ключ.

7. Переход к новому ключу.

8. Переход к следующей строке.

9. Результаты должны содержаться в списке ключей. Если совпадений не найдено, в список ключей ничего не добавляется.

Операции с ключами.

С базой ключей могут осуществляться следующие действия:

- - добавить новый ключ;
- - удалить одну запись;
- - изменить активную запись;
- - очистить всю таблицу введенных ключей.

Примечание. Под словами «работа с таблицей ключей» имеются в виду ключи, введенные для использования в двух методах (гаммирования и таблица Виженера).

Ключи для перестановки.

В каждый момент времени в системе может быть только один текущий ключ для перестановки.

Правила ввода ключа для перестановки:

1. при переключении в списке поддерживаемых системой методов шифрования на пункт «Перестановка» вызывается окно ввода ключа перестановки. Окно состоит из двух кнопок (ОТМЕНЫ и ВЫХОДА без изменений, и кнопки ENTER - подтверждение установленной длины ключа) и окна задания длины ключа для перестановки;

2. в окне задания длины ключа необходимо выбрать необходимую длину (параметры заменяются в пределах 1...9) и подтвердить желание использовать ключ именно такой длины;

3. после подтверждения в окне высветятся кнопки с цифрами на лицевой стороне (в количестве, равном длине ключа), при нажатии на кнопку происходит фиксация кнопки (ее обесцвечивание) для невозможности ее дальнейшего использования (так как все цифры в ключе перестановки должны быть неповторяющимися);

4. после перебора всех кнопок система запоминает введенный ключ, выводит его в поле ввода ключей и выходит из окна ввода ключа перестановки в окно основной программы.

Задание 1

1. Ознакомиться с описанием лабораторной работы и заданием.

2. Для одноалфавитного метода с фиксированным смещением определить установленное в программе смещение. Для этого следует:

просмотреть предварительно созданный с помощью редактора свой текстовый файл;

- выполнить для этого файла шифрование;
- просмотреть в редакторе зашифрованный файл;
- просмотреть гистограммы исходного и зашифрованного текстов;
- описать гистограммы (в чем похожи, чем отличаются) и определить с каким смещением было выполнено шифрование;
- расшифровать зашифрованный текст:

- с помощью программы, после чего проверить в редакторе правильность расшифрования,

- вручную с помощью гистограмм; описать и объяснить процесс дешифрования.

В отчете для каждого метода шифрования описывается последовательность выполняемых действий, имена всех использованных файлов, полученные гистограммы, указывается найденное смешение, описывается процесс дешифрования.

Преподавателю предоставляется отчет о проделанной работе и все использованные и созданные файлы.

3. Для одноалфавитного метода с задаваемым смещением (шифр Цезаря) следует:

- выполнить шифрование с произвольным смещением для своего исходного текста;
- просмотреть и описать гистограммы исходного и зашифрованного текстов, определить смещение для нескольких символов;
- расшифровать текст с помощью программы;
- дешифровать зашифрованный шифром Цезаря текст с помощью программы методом подбора смещения; указать, с каким смещением был зашифрован файл.

4. Для метода перестановки символов дешифровать зашифрованный файл.

Для этого необходимо определить закон перестановки символов открытого текста. Создайте небольшой файл длиной в несколько слов с известным вам текстом, зашифруйте его, просмотрите гистограммы (опишите их; ответьте, можно ли извлечь из них полезную для дешифрации информацию). Сравните (с помощью редактора) ваш исходный и зашифрованный тексты и определите закон перестановки символов.

Дешифруйте файл:

- вручную (объясните ваши действия);
- с помощью программы.

5. Для инверсного кодирования (по дополнению до 255):

- выполните шифрование для своего произвольного файла;
- просмотрите гистограммы исходного и зашифрованного текстов, опишите гистограммы и определите смещение для нескольких символов;
- дешифруйте зашифрованный текст, проверьте в редакторе правильность дешифрования.

6. Для многоалфавитного шифрования с фиксированным ключом определите, сколько одноалфавитных методов и с каким смещением используется в программе. Для этого нужно создать свой файл, состоящий из строки одинаковых символов, выполнитъ для нею шифрование и по гистограмме определить способ шифрования и набор смещений.

7. Для многоалфавитного шифрования с ключом фиксированной длины:

- выполните шифрование и определите по гистограмме, какое смещение получает каждый символ для файла, состоящего из строки одинаковых символов;
- выполните шифрование и расшифрование для файла произвольного текста;
- просмотрите и опишите гистограммы исходного и зашифрованного текстов; ответьте, какую информацию можно получить из гистограмм.

8. Для многоалфавитного шифрования с произвольным паролем задание полностью аналогично п. 7.

9. Привести в отчете ответы на контрольные вопросы в соответствии с номером варианта, указанным преподавателем.

Задание 2

1. Ознакомиться с описанием лабораторной работы и заданием.
2. Выполнить настройку программы: выбрать метод шифрования, ввести ключи для всех методов, ввести вероятное слово, осуществить все остальные системные настройки.
3. Для метода замены (одноалфавитного метода):

- выбрать данный алгоритм в списке доступных методов шифрования;
- установить необходимое смешение;
- открыть произвольный файл;
- просмотреть содержимое исходного файла;
- выполнить для этого файла шифрование (при необходимости можно задать имя зашифрованного файла);
- просмотреть в редакторе зашифрованный файл;
- ввести вероятное слово;
- ввести вероятную длину ключа (кроме метода замены);
- подобрать ключ;
- выполнить расшифрование со всеми найденными ключами;
- найти в каком-либо из расшифрованных файлов правильно расшифрованное ключевое слово;
- расшифровать файл исходным ключом;
- проверить результат.

4. Для метода перестановки:

- выбрать метод перестановки;
- в открывшемся окне ввода ключа перестановки символов указать сначала длину этого ключа, а затем из появившихся кнопок составить необходимую комбинацию для ключа, нажимая на кнопки в заданном порядке; при этом уже использованные кнопки становятся недоступными для предотвращения их повторного ввода;
- далее действия полностью соответствуют наложенным в п. 3.

5. Для метода гаммирования:

- выбрать метод;
- ввести ключ;
- полностью повторить п. 3.

6.Для таблицы Виженера все действия повторяются из п. 5 (метод гаммирования).

В отчете для каждого метода шифрования описывается последовательность выполняемых действий, указываются имена всех использованных файлов, исходные и найденные ключи, описывается процесс дешифрования.

Преподавателю предоставляется отчет о проделанной работе и все использованные файлы.

7. Привести в отчете ответы на контрольные вопросы в соответствии с номером варианта, указанным преподавателем.

Контрольные вопросы

1. Какие вы знаете методы криптографической защиты файлов?
2. В чем преимущества и недостатки одноалфавитных методов?
3. Если необходимо зашифровать текст, содержащий важную информацию, какой метод из рассмотренных вы выберете? Обоснуйте свой выбор.
4. Целесообразно ли повторно применять для уже зашифрованного текста: а) метод многоалфавитного шифрования; б) метод Цезаря?
5. Чем отличается псевдооткрытый текст (текст, полученный при расшифровке по ложному ключу) от настоящего открытого текста?
6. Как зависит время вскрытия шифра описанным выше способом подбора ключей от длины вероятного слова?
7. Зависит ли время вскрытия шифра гаммирования (или таблицы Виженера) от мощности алфавита гаммы?
8. В чем недостатки метода дешифрования с использованием протяжки вероятного слова?

Литература

1. Бабаш А.В., Шанкин Г.П. Крипотграфия. /Под редакцией В.П. Шерстюка, Э.А. Применко. – М.: СОЛОН-ПРЕСС, 2007.

2. Козлов С.Н. Защита информации. Устройства несанкционированного съема информации и борьба с ними. – М.: Академический проект, 2017.

3. Баранова Е.К., Бабаш А.В. Криптографические методы защиты информации. Лабораторный практикум. Учебное пособие – М.: КНОРУС, 2017.

4. Болелов Э.А. Криптографические методы защиты информации. Ч. 1. Симметричные криптосистемы. – М.:МГТУ ГА, 2011.

5. Болелов Э.А. Криптографические методы защиты информации. Ч. 2. Асимметричные криптосистемы. – М.:МГТУ ГА, 2011.

Лабораторная работа №2.

Исследование процесса шифрования и расшифрования ассиметричных криптосистем

Цель работы - Ознакомление с принципами шифрования, используемыми в алгоритме симметричного шифрования AES RIJNDAEL

Описание лабораторной работы

Демонстрационная версия криптостойкого блочного алгоритма Rijndael.

Состояние, ключ шифрования и число циклов. Rijndael – итеративный блочный шифр, имеющий переменную длину блоков и различные длины ключей. Длина ключа и длина блока могут быть независимо друг от друга 128, 192 или 256 бит.

Разнообразные преобразования работают с промежуточным результатом, называемым Состоянием (State). Состояние можно представить в виде прямоугольного массива байтов. Этот массив имеет четыре строки, а число столбцов обозначено как Nb и равно длине блока, делённой на 32.

Ключ шифрования также представлен в виде прямоугольного массива с четырьмя строками. Число столбцов обозначено как Nk и равно длине ключа, делённой на 32 (рис. 1).

В некоторых случаях ключ шифрования изображается в виде линейного массива четырёхбайтовых слов. Слова состоят из четырёх байтов, которые находятся в одном столбце (при представлении в виде прямоугольного массива).

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$	$a_{0,4}$	$a_{0,5}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	$a_{1,4}$	$a_{1,5}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$	$a_{2,4}$	$a_{2,5}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$	$a_{3,4}$	$a_{3,5}$

$k_{0,0}$	$k_{0,1}$	$k_{0,2}$	$k_{0,3}$
$k_{1,0}$	$k_{1,1}$	$k_{1,2}$	$k_{1,3}$
$k_{2,0}$	$k_{2,1}$	$k_{2,2}$	$k_{2,3}$
$k_{3,0}$	$k_{3,1}$	$k_{3,2}$	$k_{3,3}$

Рис. 1. Пример представления Состояния (Nb=6) и Ключа шифрования (Nk=4)

Входные данные для шифра обозначаются как байты состояния в порядке $a_{0,0}, a_{1,0}, a_{2,0}, a_{3,0}, a_{0,1}, a_{1,1}, a_{2,1}, a_{3,1} \ldots$. После завершения действия шифра выходные данные получаются из байтов состояния в том же порядке.

Число циклов, обозначенное Nr, зависит от значений Nb и Nk. (табл. 1).

Таблица 1

Nr	Nb = 4	Nb = 6	Nb = 8
Nk = 4	10	12	14
Nk = 6	12	12	14
Nk = 8	14	14	14

Цикловое преобразование.

Цикловое преобразование состоит из четырёх различных преобразований. На языке псевдо-Си оно имеет следующий вид:

```
Round (State, RoundKey)
{
    ByteSub(State); // замена байт
    ShiftRow(State); // сдвиг строк
    MixColumn(State); // замешивание столбцов
    AddRoundKey(State, RoundKey); // добавление циклового ключа
}
```

Последний цикл шифра немного отличается:

```
FinalRound(State, RoundKey)
{
    ByteSub(State); // замена байт ShiftRow(State); // сдвиг
строк
     AddRoundKey(State, RoundKey); // добавление циклового ключа
}
```

Отметим, что последний цикл отличается от простого цикла только отсутствием замешивания столбцов. Каждое из приведённых преобразований подробно рассмотрено далее.

Замена байт (ByteSub). Преобразование ByteSub – нелинейная замена байт, выполняемая независимо с каждым байтом состояния (рис. 2).

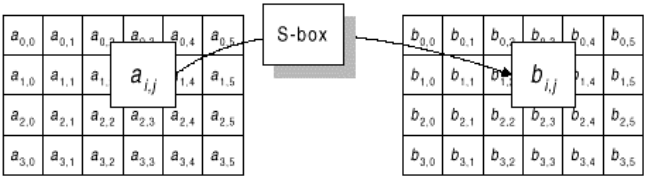

Рис.2. ByteSub действует на каждый байт состояния

Замена происходит по массиву SboxE при шифровании и по массиву SboxD при расшифровании, причём SboxD[SboxE[a]] = a. На языке псевдо-Си это выглядит следующим образом:

```
SboxE = {
0x63, 0x7C, 0x77, 0x7B, 0xF2, 0x6B, 0x6F, 0xC5, 0x30,
0x01, 0x67, 0x2B, 0xFE, 0xD7, 0xAB, 0x76,
0xCA, 0x82, 0xC9, 0x7D, 0xFA, 0x59, 0x47, 0xF0, 0xAD,
0xD4, 0xA2, 0xAF, 0x9C, 0xA4, 0x72, 0xC0,
0xB7, 0xFD, 0x93, 0x26, 0x36, 0x3F, 0xF7, 0xCC, 0x34,
0xA5, 0xE5, 0xF1, 0x71, 0xD8, 0x31, 0x15,
0x04, 0xC7, 0x23, 0xC3, 0x18, 0x96, 0x05, 0x9A, 0x07,
0x12, 0x80, 0xE2, 0xEB, 0x27, 0xB2, 0x75,
0x09, 0x83, 0x2C, 0x1A, 0x1B, 0x6E, 0x5A, 0xA0, 0x52,
0x3B, 0xD6, 0xB3, 0x29, 0xE3, 0x2F, 0x84, 0x53, 0xD1,
0x00, 0xED, 0x20, 0xFC, 0xB1, 0x5B, 0x6A, 0xCB, 0xBE,
0x39, 0x4A, 0x4C, 0x58, 0xCF, 0xD0, 0xEF, 0xAA, 0xFB,
0x43, 0x4D, 0x33, 0x85, 0x45, 0xF9, 0x02, 0x7F, 0x50,
0x3C, 0x9F, 0xA8, 0x51, 0xA3, 0x40, 0x8F, 0x92, 0x9D,
0x38, 0xF5, 0xBC, 0xB6, 0xDA, 0x21, 0x10, 0xFF, 0xF3,
0xD2, 0xCD, 0x0C, 0x13, 0xEC, 0x5F, 0x97, 0x44, 0x17,
0xC4, 0xA7, 0x7E, 0x3D, 0x64, 0x5D, 0x19, 0x73, 0x60,
0x81, 0x4F, 0xDC, 0x22, 0x2A, 0x90, 0x88, 0x46, 0xEE,
0xB8, 0x14, 0xDE, 0x5E, 0x0B, 0xDB, 0xE0, 0x32, 0x3A,
0x0A, 0x49, 0x06, 0x24, 0x5C, 0xC2, 0xD3, 0xAC, 0x62,
0x91, 0x95, 0xE4, 0x79, 0xE7, 0xC8, 0x37, 0x6D, 0x8D,
0xD5, 0x4E, 0xA9, 0x6C, 0x56, 0xF4, 0xEA, 0x65, 0x7A,
0xAE, 0x08,
0xBA, 0x78, 0x25, 0x2E, 0x1C, 0xA6, 0xB4, 0xC6, 0xE8,
0xDD, 0x74, 0x1F, 0x4B, 0xBD, 0x8B, 0x8A,
0x70, 0x3E, 0xB5, 0x66, 0x48, 0x03, 0xF6, 0x0E, 0x61,
0x35, 0x57, 0xB9, 0x86, 0xC1, 0x1D, 0x9E,
0xE1, 0xF8, 0x98, 0x11, 0x69, 0xD9, 0x8E, 0x94, 0x9B,
0x1E, 0x87, 0xE9, 0xCE, 0x55, 0x28, 0xDF,
```

```
0x8C, 0xA1, 0x89, 0x0D, 0xBF, 0xE6, 0x42, 0x68, 0x41,
0x99, 0x2D, 0x0F, 0xB0, 0x54, 0xBB, 0x16
};
SboxD = {
0x52, 0x09, 0x6A, 0xD5, 0x30, 0x36, 0xA5, 0x38, 0xBF,
0x40, 0xA3, 0x9E, 0x81, 0xF3, 0xD7, 0xFB,
0x7C, 0xE3, 0x39, 0x82, 0x9B, 0x2F, 0xFF, 0x87, 0x34,
0x8E, 0x43, 0x44, 0xC4, 0xDE, 0xE9, 0xCB,
0x54, 0x7B, 0x94, 0x32, 0xA6, 0xC2, 0x23, 0x3D, 0xEE,
0x4C, 0x95, 0x0B, 0x42, 0xFA, 0xC3, 0x4E,
0x08, 0x2E, 0xA1, 0x66, 0x28, 0xD9, 0x24, 0xB2, 0x76,
0x5B, 0xA2, 0x49, 0x6D, 0x8B, 0xD1, 0x25,
0x72, 0xF8, 0xF6, 0x64, 0x86, 0x68, 0x98, 0x16, 0xD4,
0xA4, 0x5C, 0xCC, 0x5D, 0x65, 0xB6, 0x92,
0x6C, 0x70, 0x48, 0x50, 0xFD, 0xED, 0xB9, 0xDA, 0x5E,
0x15, 0x46, 0x57, 0xA7, 0x8D, 0x9D, 0x84,
0x90, 0xD8, 0xAB, 0x00, 0x8C, 0xBC, 0xD3, 0x0A, 0xF7,
0xE4, 0x58, 0x05, 0xB8, 0xB3, 0x45, 0x06,
0xD0, 0x2C, 0x1E, 0x8F, 0xCA, 0x3F, 0x0F, 0x02, 0xC1,
0xAF, 0xBD, 0x03, 0x01, 0x13, 0x8A, 0x6B,
0x3A, 0x91, 0x11, 0x41, 0x4F, 0x67, 0xDC, 0xEA, 0x97,
0xF2, 0xCF, 0xCE, 0xF0, 0xB4, 0xE6, 0x73,
0x96, 0xAC, 0x74, 0x22, 0xE7, 0xAD, 0x35, 0x85, 0xE2,
0xF9, 0x37, 0xE8, 0x1C, 0x75, 0xDF, 0x6E,
0x47, 0xF1, 0x1A, 0x71, 0x1D, 0x29, 0xC5, 0x89, 0x6F,
0xB7, 0x62, 0x0E, 0xAA, 0x18, 0xBE, 0x1B,
0xFC, 0x56, 0x3E, 0x4B, 0xC6, 0xD2, 0x79, 0x20, 0x9A,
0xDB, 0xC0, 0xFE, 0x78, 0xCD, 0x5A, 0xF4,
0x1F, 0xDD, 0xA8, 0x33, 0x88, 0x07, 0xC7, 0x31, 0xB1,
0x12, 0x10, 0x59, 0x27, 0x80, 0xEC, 0x5F,
0x60, 0x51, 0x7F, 0xA9, 0x19, 0xB5, 0x4A, 0x0D, 0x2D,
0xE5, 0x7A, 0x9F, 0x93, 0xC9, 0x9C, 0xEF,
0xA0, 0xE0, 0x3B, 0x4D, 0xAE, 0x2A, 0xF5, 0xB0, 0xC8,
0xEB, 0xBB, 0x3C, 0x83, 0x53, 0x99, 0x61,
0x17, 0x2B, 0x04, 0x7E, 0xBA, 0x77, 0xD6, 0x26, 0xE1,
0x69, 0x14, 0x63, 0x55, 0x21, 0x0C, 0x7D
};
```

Преобразование сдвига строк (ShiftRow).

Последние три строки состояния циклически сдвигаются на различное число байт. Строка 1 сдвигается на C1 байт, строка 2 – на C2 байт и строка 3 – на C3 байт. Значения сдвигов C1, C2, C3 зависят от длины блока Nb. Их величины приведены в табл. 2.

Таблица 2

Nb	C1	C2	C3
4	1	2	3
6	1	2	3
8	1	3	4

Операция сдвига последних трёх строк состояния на определённую величину обозначена как ShiftRow (State). На рисунке 3 показано влияние преобразования на состояние.

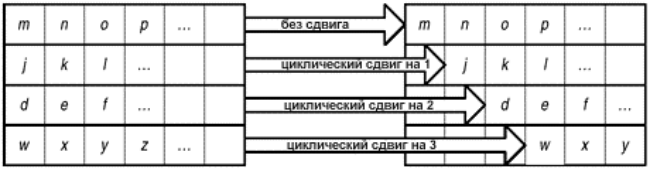

Рис. 3. Схема преобразования ShiftRow

При расшифровании происходит сдвиг на то же число элементов в обратном направлении.

Преобразование замешивания столбцов (MixColumn).

Преобразование представляет собой умножение состояния на матрицу ME при шифровании или матрицу MD при расшифровании:

$$ME \otimes State$$

$$MD \otimes State$$

$$\begin{vmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{vmatrix} \otimes \begin{vmatrix} b1 & b5 & b9 & b13 \\ b2 & b6 & b10 & b14 \\ b3 & b7 & b11 & b15 \\ b4 & b8 & b12 & b16 \end{vmatrix}$$

```
b1 = (b1*2) XOR (b2*3) XOR (b3*1) XOR (b4*1)
```

Умножение двух байт выполняется по следующему алгоритму:

- если один байт равен 0, результатом будет 0;
- если один байт равен 1, результатом будет другой байт;
- в остальных случаях происходит замена каждого байта по таблице L.

Заменённые байты складываются, при необходимости вычитается 255 для попадания в интервал [0, 255] и происходит замена по таблице E, что и даёт результат. На языке псевдо-Си таблицы L и E имеют следующий вид:

```
L = {
    , 0x00, 0x19, 0x01, 0x32, 0x02, 0x1A, 0xC6, 0x4B,
0xC7, 0x1B, 0x68, 0x33, 0xEE, 0xDF, 0x03,
```

```
0x64, 0x04, 0xE0, 0x0E, 0x34, 0x8D, 0x81, 0xEF, 0x4C,
0x71, 0x08, 0xC8, 0xF8, 0x69, 0x1C, 0xC1,
0x7D, 0xC2, 0x1D, 0xB5, 0xF9, 0xB9, 0x27, 0x6A, 0x4D,
0xE4, 0xA6, 0x72, 0x9A, 0xC9, 0x09, 0x78,
0x65, 0x2F, 0x8A, 0x05, 0x21, 0x0F, 0xE1, 0x24, 0x12,
0xF0, 0x82, 0x45, 0x35, 0x93, 0xDA, 0x8E,
0x96, 0x8F, 0xDB, 0xBD, 0x36, 0xD0, 0xCE, 0x94, 0x13,
0x5C, 0xD2, 0xF1, 0x40, 0x46, 0x83, 0x38,
0x66, 0xDD, 0xFD, 0x30, 0xBF, 0x06, 0x8B, 0x62, 0xB3,
0x25, 0xE2, 0x98, 0x22, 0x88, 0x91, 0x10,
0x7E, 0x6E, 0x48, 0xC3, 0xA3, 0xB6, 0x1E, 0x42, 0x3A,
0x6B, 0x28, 0x54, 0xFA, 0x85, 0x3D, 0xBA,
0x2B, 0x79, 0x0A, 0x15, 0x9B, 0x9F, 0x5E, 0xCA, 0x4E,
0xD4, 0xAC, 0xE5, 0xF3, 0x73, 0xA7, 0x57,
0xAF, 0x58, 0xA8, 0x50, 0xF4, 0xEA, 0xD6, 0x74, 0x4F,
0xAE, 0xE9, 0xD5, 0xE7, 0xE6, 0xAD, 0xE8,
0x2C, 0xD7, 0x75, 0x7A, 0xEB, 0x16, 0x0B, 0xF5, 0x59,
0xCB, 0x5F, 0xB0, 0x9C, 0xA9, 0x51, 0xA0,
0x7F, 0x0C, 0xF6, 0x6F, 0x17, 0xC4, 0x49, 0xEC, 0xD8,
0x43, 0x1F, 0x2D, 0xA4, 0x76, 0x7B, 0xB7,
0xCC, 0xBB, 0x3E, 0x5A, 0xFB, 0x60, 0xB1, 0x86, 0x3B,
0x52, 0xA1, 0x6C, 0xAA, 0x55, 0x29, 0x9D,
0x97, 0xB2, 0x87, 0x90, 0x61, 0xBE, 0xDC, 0xFC, 0xBC,
0x95, 0xCF, 0xCD, 0x37, 0x3F, 0x5B, 0xD1,
0x53, 0x39, 0x84, 0x3C, 0x41, 0xA2, 0x6D, 0x47, 0x14,
0x2A, 0x9E, 0x5D, 0x56, 0xF2, 0xD3, 0xAB,
0x44, 0x11, 0x92, 0xD9, 0x23, 0x20, 0x2E, 0x89, 0xB4,
0x7C, 0xB8, 0x26, 0x77, 0x99, 0xE3, 0xA5,
0x67, 0x4A, 0xED, 0xDE, 0xC5, 0x31, 0xFE, 0x18, 0x0D,
0x63, 0x8C, 0x80, 0xC0, 0xF7, 0x70, 0x07
};
E = {
0x01, 0x03, 0x05, 0x0F, 0x11, 0x33, 0x55, 0xFF, 0x1A,
0x2E, 0x72, 0x96, 0xA1, 0xF8, 0x13, 0x35,
0x5F, 0xE1, 0x38, 0x48, 0xD8, 0x73, 0x95, 0xA4, 0xF7,
0x02, 0x06, 0x0A, 0x1E, 0x22, 0x66, 0xAA,
0xE5, 0x34, 0x5C, 0xE4, 0x37, 0x59, 0xEB, 0x26, 0x6A,
0xBE, 0xD9, 0x70, 0x90, 0xAB, 0xE6, 0x31,
0x53, 0xF5, 0x04, 0x0C, 0x14, 0x3C, 0x44, 0xCC, 0x4F,
0xD1, 0x68, 0xB8, 0xD3, 0x6E, 0xB2, 0xCD,
0x4C, 0xD4, 0x67, 0xA9, 0xE0, 0x3B, 0x4D, 0xD7, 0x62,
0xA6, 0xF1, 0x08, 0x18, 0x28, 0x78, 0x88,
```

```
0x83, 0x9E, 0xB9, 0xD0, 0x6B, 0xBD, 0xDC, 0x7F, 0x81,
0x98, 0xB3, 0xCE, 0x49, 0xDB, 0x76, 0x9A,
0xB5, 0xC4, 0x57, 0xF9, 0x10, 0x30, 0x50, 0xF0, 0x0B,
0x1D, 0x27, 0x69, 0xBB, 0xD6, 0x61, 0xA3,
0xFE, 0x19, 0x2B, 0x7D, 0x87, 0x92, 0xAD, 0xEC, 0x2F,
0x71, 0x93, 0xAE, 0xE9, 0x20, 0x60, 0xA0,
0xFB, 0x16, 0x3A, 0x4E, 0xD2, 0x6D, 0xB7, 0xC2, 0x5D,
0xE7, 0x32, 0x56, 0xFA, 0x15, 0x3F, 0x41,
0xC3, 0x5E, 0xE2, 0x3D, 0x47, 0xC9, 0x40, 0xC0, 0x5B,
0xED, 0x2C, 0x74, 0x9C, 0xBF, 0xDA, 0x75,
0x9F, 0xBA, 0xD5, 0x64, 0xAC, 0xEF, 0x2A, 0x7E, 0x82,
0x9D, 0xBC, 0xDF, 0x7A, 0x8E, 0x89, 0x80,
0x9B, 0xB6, 0xC1, 0x58, 0xE8, 0x23, 0x65, 0xAF, 0xEA,
0x25, 0x6F, 0xB1, 0xC8, 0x43, 0xC5, 0x54,
0xFC, 0x1F, 0x21, 0x63, 0xA5, 0xF4, 0x07, 0x09, 0x1B,
0x2D, 0x77, 0x99, 0xB0, 0xCB, 0x46, 0xCA,
0x45, 0xCF, 0x4A, 0xDE, 0x79, 0x8B, 0x86, 0x91, 0xA8,
0xE3, 0x3E, 0x42, 0xC6, 0x51, 0xF3, 0x0E,
0x12, 0x36, 0x5A, 0xEE, 0x29, 0x7B, 0x8D, 0x8C, 0x8F,
0x8A, 0x85, 0x94, 0xA7, 0xF2, 0x0D, 0x17,
0x39, 0x4B, 0xDD, 0x7C, 0x84, 0x97, 0xA2, 0xFD, 0x1C,
0x24, 0x6C, 0xB4, 0xC7, 0x52, 0xF6, 0x01
};
```

Добавление циклового ключа.

Цикловой ключ добавляется к состоянию посредством простого EXOR (рис. 4). Цикловой ключ вырабатывается из ключа шифрования посредством алгоритма выработки ключей (key schedule). Длина циклового ключа равна длине блока Nb.

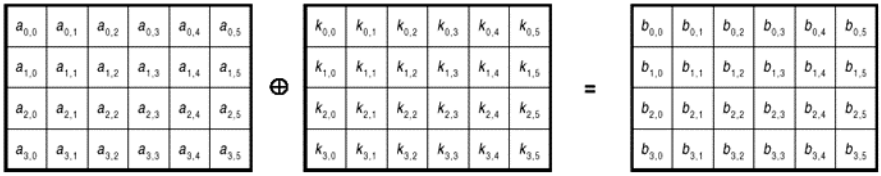

Рис. 4. Операция добавления циклового ключа

При шифровании части расширенного ключа выбираются от начала к концу, при расшифровании – от конца к началу.

Расширение ключа (Key Expansion).

Расширенный ключ представляет собой линейный массив четырёх байтовых слов и обозначается W[Nb*(Nr+1)]. Первые Nk слов содержат ключ шифрования. Все остальные слова определяются рекурсивно из слов с меньшими индексами. Алгоритм выработки ключей зависит от величины Nk. Ниже приведена версия для Nk ≤ 6 и версия для Nk > 6.

- для Nk < 6 или Nk = 6

```
KeyExpansion(CipherKey,W)
{
        for (i = 0; i < Nk; i++) W[i] = CipherKey[i];
                for (j = Nk; j < Nb*(Nk+1); j+=Nk)
                 {
                        W[j] = W[j-Nk] ^ SubByte( Rotl( W[j-1] ) ) ^
                        Rcon[j/Nk];
                        for (i = 1; i < Nk && i+j < Nb*(Nr+1); i++)
                        W[i+j] = W[i+j-Nk] ^ W[i+j-1];
                 }
}
```

- для Nk > 6

```
KeyExpansion(CipherKey,W)
{
        for (i=0; i<Nk; i++) W[i]=CipherKey[i];
                for (j=Nk; j<Nb*(Nk+1); j+=Nk)
                 {
                        W[j] = W[j-Nk] ^ SubByte(Rotl(W[j-1])) ^
                        Rcon[j/Nk];
                        for (i=1; i<4; i++) W[i+j] = W[i+j-Nk] ^
                        W[i+j-1];
                        W[j+4] = W[j+4-Nk] ^ SubByte(W[j+3]);
                        for (i=5; i<Nk; i++) W[i+j] = W[i+j-Nk] ^
                        W[i+j-1];
                 }
}
```

Отличие для схемы при Nk > 6 состоит в применении SubByte для каждого четвёртого байта из Nk.

Цикловая константа не зависит от Nk и определяется следующим образом:

Rcon[i] = (RC[i], '00' , '00' , '00'), где
RC[0]='01'

RC[i]=xtime(Rcon[i-1])

Шифрование.

Шифр Rijndael включает следующие преобразования:

- начальное добавление циклового ключа;
- Nr-1 циклов;
- Заключительный цикл.

На языке псевдо-Си выглядит следующим образом:

```
Rijndael (State, CipherKey)
{
        KeyExpansion(CipherKey, ExpandedKey); // Расширение ключа
        AddRoundKey(State, ExpandedKey); // Добавление циклового ключа
        For (i=1; i<Nr; i++) Round(State,ExpandedKey+Nb*i); // циклы
        FinalRound(State, ExpandedKey+Nb*Nr); // заключительный цикл
}
```

Если предварительно выполнена процедура расширения ключа, то процедура будет выглядеть следующим образом:

```
Rijndael (State, CipherKey)
{
        AddRoundKey(State, ExpandedKey);
        For (i=1; i<Nr; i++) Round(State,ExpandedKey+Nb*i);
        FinalRound(State, ExpandedKey+Nb*Nr);
}
```

Описание демонстрационной программы.

Программа выполненная на языке C# и состоит из двух элементов – файла Rijndael.dll, содержащего реализацию алгоритма шифрования, и демонстрационного приложения RijndaelDemo.exe. Для работы приложения необходима ОС Windows с установленным .NET Framework v1.1.

В основном окне демонстрационной программы задаются длина ключа, длина блока, можно посмотреть расширенный ключ шифрования, вычисляемый в соответствии с заданным ключом шифрования. (рис. 5, 6).

Можно подробно рассмотреть действие всех цикловых преобразований (ByteSub, ShiftRow, MixColumn, AddRoundKey) как при шифровании, так и при расшифровании (рис. 7, 8).

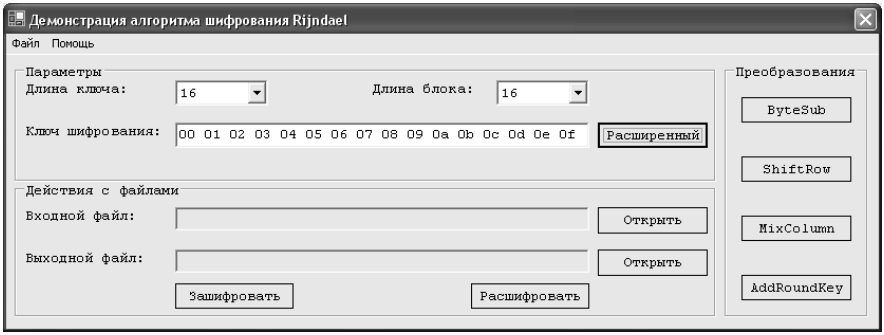

Рис. 5. Главное окно демонстрационной программы

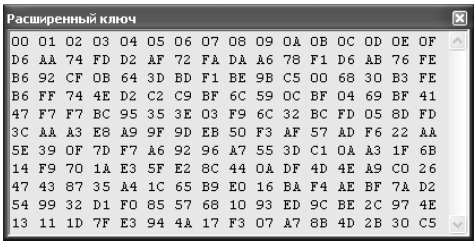

Рис. 6. Окно расширенного ключа

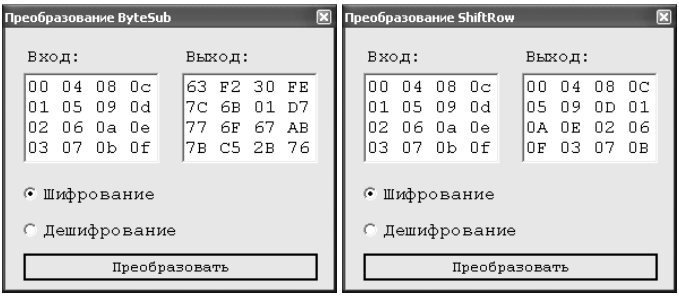

Рис. 7. Окна преобразований ByteSub и ShiftRow

При шифровании предлагается выбрать исходный файл и файл, куда будет помещён результат шифрования, при расшифровании соответственно зашифрованный файл и файл, предназначенный для помещения результата расшифрования. В процессе используются, указанные в главном окне программы, ключ шифрования и длины ключа и блока.

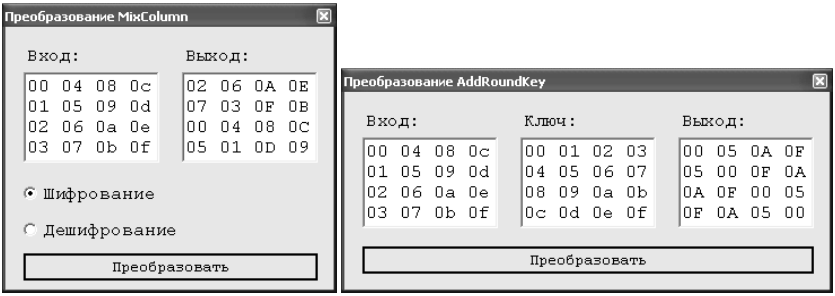

Рис. 8**.** Окна преобразований MixColumn и AddRoundKey

Задание

1. Ознакомиться со сведениями о программе RijndaelDemo. Запустить модуль RijndaelDemo.exe.
2. Изучить на примере обычных текстовых файлов способы шифрования и расшифрования с помощью алгоритма Rijndael. Подробно рассмотреть действие всех цикловых преобразований (ByteSub, ShiftRow, MixColumn, AddRoundKey), как при шифровании, так и расшифровании. Исходный текст для шифрования может быть подготовлен заранее и сохранен в файле *.txt.
3. Сохранить в отчёте экранные формы, демонстрирующие процесс шифрования и расшифрования информации, проанализировать полученные результаты.
4. Включить в отчёт о лабораторной работе ответы на контрольные вопросы.

Контрольные вопросы

1. Сравните основные характеристики алгоритмов Rijndael и ГОСТ 28147-89
2. Сравните основные характеристики алгоритмов Rijndael и DES
3. Опишите структуру сети Фейстеля
4. Приведите обобщённые схемы шифрования данных с помощью алгоритма Rijndael и ГОСТ 28147-89. Дайте их сравнительный анализ
5. Сравните один раунд шифрования данных с помощью алгоритма

Rijndael и ГОСТ 28147-89

6. Сравните эквивалентность прямого и обратного преобразований в алгоритмах Rijndael и ГОСТ 28147-89

7. Сравните выработку ключевой информации в алгоритмах Rijndael и ГОСТ 28147-89

8. Сравните алгоритмы Rijndael и ГОСТ 28147-89 по показателям диффузии

9. Сравните алгоритмы Rijndael и ГОСТ 28147-89 по показателям стойкости

10. Сравните алгоритмы Rijndael и ГОСТ 28147-89 по показателям производительности и удобству реализации

Литература

1. Бабаш А.В., Шанкин Г.П. Криптография. /Под редакцией В.П. Шерстюка, Э.А. Применко. – М.: СОЛОН-ПРЕСС, 2007.

2. Козлов С.Н. Защита информации. Устройства несанкционированного съема информации и борьба с ними. – М.: Академический проект, 2017.

3. Баранова Е.К., Бабаш А.В. Криптографические методы защиты информации. Лабораторный практикум. Учебное пособие – М.: КНОРУС, 2017.

4. Болелов Э.А. Криптографические методы защиты информации. Ч. 1. Симметричные криптосистемы. – М.:МГТУ ГА, 2011.

5. Болелов Э.А. Криптографические методы защиты информации. Ч. 2. Асимметричные криптосистемы. – М.:МГТУ ГА, 2011.

Лабораторная работа №3.

Исследование алгоритмов электронной подписи

Цель работы - ознакомление с принципами защищённого электронного документооборота в телекоммуникационных сетях и алгоритмами постановки электронной цифровой подписи (ЭЦП).

Описание лабораторной работы

Электронная цифровая подпись.

При обмене электронными документами по сети связи существенно снижаются затраты на обработку и хранение документов, ускоряется их поиск. Но при этом возникает проблема аутентификации автора документа и самого документа, т.е. установления подлинности автора и отсутствия изменений в полученном документе.

При обработке документов в электронной форме совершенно непригодны традиционные способы установления подлинности по рукописной подписи и оттиску печати на бумажном документе. Принципиально новым решением является электронная цифровая подпись.

Первая схема ЭЦП – RSA была разработана ещё в конце 1970-х годов. Однако проблема подтверждения авторства стала актуальной настолько, что потребовалось установление стандарта, только в 1990-х годах, во время взрывного роста глобальной сети Интернет и массового распространения электронной торговли и оказания услуг. Именно по указанной причине стандарты ЭЦП в России и США были приняты практически одновременно, в 1994 году.

Из предложенных криптологами схем ЭЦП наиболее удачными оказались RSA и схема Эль-Гамаля. Но первая из них была запатентована в США и ряде других стран (патент на RSA прекратил своё действие совсем недавно). У второй схемы существует большое количество её возможных модификаций, и все их запатентовать весьма затруднительно. Именно по этой причине схема ЭЦП Эль-Гамаля осталась по большей части свободной от патентов. Кроме того, эта схема имеет и определённые практические преимущества: размер блоков, которыми оперируют алгоритмы, и соответственно размер ЭЦП в ней оказались значительно меньше, чем в RSA, при той же самой стойкости. Именно поэтому стандарты ЭЦП России и США базируются на схеме Эль-

Гамаля.

Законы об ЭЦП сегодня имеют уже более 60-ти государств. В этом списке значится и Россия. Принятый Федеральный закон от 6 апреля 2011 г. № 63-ФЗ «Об электронной подписи» должен оказать стимулирующее воздействие на развитие отечественной электронной коммерции, особенно если в соответствие с ним будут своевременно приведены иные нормативно-правовые акты.

- Правительство РФ финансово поддерживает осуществление федеральной целевой программы «Электронная Россия»;
- принят закон «О внесении изменения в статью 80 части первой Налогового кодекса Российской Федерации»;
- ещё в январе 2001 г. правление Пенсионного фонда постановлением «О введении в системе Пенсионного фонда РФ криптографической защиты информации и электронной цифровой подписи» регламентировало регистрацию и подключение юридических и физических лиц к системе своего электронного документооборота;
- в 2002 г. вышел приказ МНС России «Об утверждении порядка представления налоговой декларации в электронном виде по телекоммуникационным каналам связи», благодаря которому сегодня любое физическое или юридическое лицо может связаться с налоговой инспекцией, используя защищённую электронную почту;
- в 2004 г. были утверждены поправки к статьям 13 и 15 закона «О бухгалтерском учёте», согласно которым бухгалтерская отчётность предприятия может вестись, храниться и предоставляться в контролирующие органы в электронном виде.

Принцип построения ЭЦП.

Асимметрия ролей отправителя и получателя в схемах ЭЦП требует наличия двух тесно связанных ключей: *секретного,* или ключа подписи, и открытого, или ключа проверки подписи.

Любая схема ЭЦП обязана определить три следующих алгоритма:

1) генерации ключевой пары для подписи и её проверки;

2) постановки подписи;

3) проверки подписи.

Стандарты России и США очень похожи, они различаются лишь некоторыми числовыми параметрами и отдельными деталями выработки ключевой пары, вычисления и проверки подписи. Действительно, оба стандарта

являются вариантами одной и той же схемы ЭЦП Эль-Гамаля.

ЭЦП используется для аутентификации текстов, передаваемых по телекоммуникационным каналам. Функционально она аналогична обычной рукописной подписи и обладает основными её достоинствами:

- удостоверяет, что подписанный текст исходит от лица, поставившего подпись;
- не даёт самому этому лицу возможность отказаться от обязательств, связанных с подписанным текстом;
- гарантирует целостность подписанного текста.

ЭЦП представляет собой относительно небольшое количество дополнительной цифровой информации, передаваемой вместе с подписываемым текстом, и включает две процедуры:

1) процедуру постановки подписи, в которой используется секретный ключ отправителя сообщения;

2) процедуру проверки подписи, в которой используется открытый ключ отправителя.

Процедура постановки подписи.

При формировании ЭЦП, отправитель, прежде всего, вычисляет хэш-функцию $h(M)$ подписываемого текста M. Вычисленное значения хэш-функции $h(M)$ представляет собой один короткий блок информации m, характеризующий весь текст M в целом. Затем значение m шифруется секретным ключом отправителя. Получаемая при этом пара чисел представляет собой ЭЦП для данного текста M.

Процедура проверки подписи.

При проверке ЭЦП, получатель сообщения снова вычисляет хэш-функцию $m = h(M)$ принятого по каналу текста M, после чего при помощи открытого ключа отправителя проверяет, соответствует ли полученная подпись вычисленному значению m хэш-функции.

Принципиальным моментом в системе ЭЦП является невозможность подделки ЭЦП пользователя без знания его секретного ключа.

Каждая подпись, как правило, содержит следующую информацию:

- дату подписи;
- срок окончания действия ключа данной подписи;
- информацию о лице, подписавшем текст;
- идентификатор подписавшего (имя открытого ключа);
- собственно цифровую подпись.

Однонаправленные хэш-функции.

Хэш-функция предназначена для сжатия подписываемого документа M до нескольких десятков или сотен бит. Хэш-функция $h(.)$ использует в качестве

аргумента сообщение M произвольной длины и возвращает хэш-значение $h(M) = H$ фиксированной длины. Обычно хэшированная информация является сжатым двоичным представлением основного сообщения произвольной длины. Следует отметить, что значение хэш-функции $h(M)$ сложным образом зависит от документа M и не позволяет восстановить сам документ M.

Хэш-функция должна удовлетворять целому ряду условий:

- должна быть чувствительна к всевозможным изменениям в тексте M;
- должна обладать свойством необратимости, т.е. задача подбора документа M_1, который обладал бы требуемым значением хэш-функции, должна быть вычислительно неразрешима;
- вероятность того, что значения хэш-функции двух различных документов совпадут, должна быть ничтожно мала.

Большинство хэш-функций строится на основе однонаправленной функции $f(.)$, которая образует выходное значение длиной n при задании двух входных значений длиной n. Этими входами являются блок исходного текста M_i и хэш-значение H_{i-1} предыдущего блока текста (рис 1).

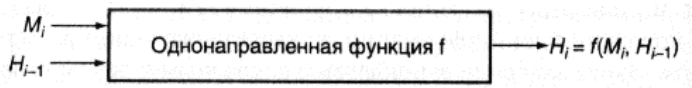

Рис. 1. Схема формирования хэш-функций

Алгоритм цифровой подписи DSA.

Алгоритм цифровой подписи DSA (Digital Signature Authorization) предложен в 1991 году в США и является развитием алгоритма цифровой подписи Эль-Гамаля.

Отправитель и получатель электронного документа используют при вычислении большие целые числа:

G, P – простые числа по *L*-бит каждое (*L* = 512 ... 1024 бит);

q – простое число длиной 160 бит делитель числа (*P* - 1).

Числа *G, P, q* являются открытыми и могут быть общими для всех пользователей сети.

Описание алгоритма

1. Отправитель выбирает случайное целое число X, $1 < X < q$. Число X является секретным ключом отправителя для формирования электронной

подписи.

2. Отправитель вычисляет значение

$Y = G^X \, mod \; P$.

Число Y является открытым ключом для проверки подписи отправителя и предаётся всем получателям документа.

3. Для того чтобы подписать документ M, отправитель хэширует его в целое хэш-значение т:

$$m = h(M),\ 1 < m < q.$$

Затем генерирует случайное целое число K, $1 < K < q$ и вычисляет число r:

$$r = \left(G^K \; mod \; P\right) \; mod \; q.$$

4. При помощи секретного ключа X отправитель вычисляет число s:

$$s = ((m + r \cdot X) / K) \; mod \; q.$$

Пара чисел *r, s* образуют цифровую подпись S = (r, s) под документом *M.*

5. Доставленное получателю сообщение вместе с подписью представляет собой тройку чисел [M, r, s]. Прежде всего получатель проверяет выполнение соотношений:

$$0 < r < q;\ 0 < s < q.$$

6. Далее получатель вычисляет значения:

$$w = 1 / s \bmod q;$$

$$m = h(M) \; - \text{хэш-значение;}$$

$$u_1 = (m \cdot w) \bmod q;$$

$$u_2 = (m \cdot w) \bmod q.$$

Затем при помощи открытого ключа *Y* вычисляется значение

$$v = ((G^{u1} \cdot Y^{12}) mod \; P) \; mod \; q$$

и проверяется выполнение равенства *v = r.* Если оно выполняется, то подпись признается подлинной, так как можно строго математически доказать, что последнее равенство будет выполняться тогда и только тогда, когда подпись *S = (r, s)* под документом *M* получена при помощи именно того секретного ключа *X,* из которого был получен открытый ключ Y.

Новые стандарты ЭЦП.

Последние достижения криптографии показали, что общая проблема логарифмирования в дискретных полях, являющаяся базой указанной схемы ЭЦП, не может считаться достаточно прочным фундаментом. Например, размеры блоков, считающиеся "безопасными", растут сравнительно быстрыми темпами. В результате это привело к тому, что стандарты ЭЦП России и США

в 2001 году были обновлены: переведены на эллиптические кривые. Наиболее употребимые алгоритмы ЭЦП приведены на рис. 2. Схемы ЭЦП при этом остались прежними, но в качестве чисел, которыми они оперируют, теперь используются не элементы конечного поля $GF(2n)$ или $GF(p)$, а эллиптические числа – решения уравнения эллиптических кривых над указанными конечными полями. Роль операции возведения числа в степень в конечном поле в обновлённых стандартах выполняет операция взятия кратной точки эллиптической кривой – «умножение» точки на целое число.

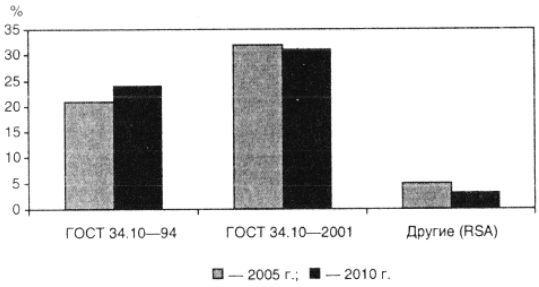

Рис. 2. Алгоритмы ЭЦП, используемые в РФ

Надлежащий выбор типа эллиптической кривой позволяет многократно усложнить задачу взлома схемы ЭЦП и уменьшить рабочий размер блоков данных. Старый российский стандарт ЭЦП оперирует 1024-битовыми блоками, а новый, основанный на эллиптических кривых, – 256-битовыми, и при этом обладает большей стойкостью.

Стойкость схемы подписи ГОСТ Р34.10–94 базируется на сложности решения задачи дискретного логарифмирования в простом поле. В настоящее время наиболее быстрым алгоритмом её решения для общего случая является алгоритм обобщённого решета числового поля.

В ГОСТ Р34.10–2001 стойкость схемы ЭЦП основана на сложности решения задачи дискретного логарифмирования в группе точек эллиптической кривой. При правильном выборе параметров кривой самыми эффективными методами её решения являются более трудоёмкие *r*- и *l*- методы Полларда. Так, по разным оценкам специалистов, трудоёмкость взлома старого и нового стандартов ЭЦП России составляет величину порядка 1026 и 1038 операций умножения в базовом поле $GF(p)$ соответственно. Очевидно, что новый стандарт более стойкий.

Задание

1. Ознакомиться с основными направлениями работ в рамках федеральной целевой программы “Электронная Россия”, а также со сведениями о порядке использования и действующих алгоритмах постановки электронной цифровой, изложенными выше.

Запустить программу labWork6.exe, предназначенную для демонстрации порядка постановки и проверки электронной цифровой подписи.

2. Сгенерировать и переслать участникам обмена ключи для шифрования исходного документа и ключи для подписания документа. Исходный текст для шифрования набирается непосредственно в окне программы.

3. Зашифровать исходное сообщение и подписать его на секретном ключе отправителя.

4. Переслать зашифрованное и подписанное сообщение получателю. Выполнить проверку правильности ЭЦП и восстановить исходный текст сообщения.

5. Сохранить в отчёте экранные формы, демонстрирующие процесс генерации и распространения ключей; процесс шифрования исходного документа и постановки ЭЦП.

6. Привести в отчёте ответы на контрольные вопросы.

Контрольные вопросы

1. В чём состоит назначение хэш-функций и какие требования предъявляются к хэш-функциям, используемым для постановки ЭЦП? Перечислите стандарты хэш-функций, действующие в Российской федерации.

2. Опишите процедуры постановки и проверки ЭЦП. Какая информация содержится в ЭЦП?

3. Приведите пример реализации алгоритма ЭЦП *(RSA, El-Gamal, DSA)*

4. Перечислите стандарты ЭЦП, действующие в Российской федерации.

5. На каких принципах основана криптостойкость современных алгоритмов ЭЦП?

Литература

1. Бабаш А.В., Шанкин Г.П. Криптография. /Под редакцией В.П. Шерстюка, Э.А. Применко. – М.: СОЛОН-ПРЕСС, 2007.

2. Козлов С.Н. Защита информации. Устройства несанкционированного съема информации и борьба с ними. – М.: Академический проект, 2017.

3. Баранова Е.К., Бабаш А.В. Криптографические методы защиты информации. Лабораторный практикум. Учебное пособие – М.: КНОРУС, 2017.

4. Болелов Э.А. Криптографические методы защиты информации. Ч. 1. Симметричные криптосистемы. – М.:МГТУ ГА, 2011.

5. Болелов Э.А. Криптографические методы защиты информации. Ч. 2. Асимметричные криптосистемы. – М.:МГТУ ГА, 2011.

Лабораторная работа №4.

Изучение метода линейного криптоанализа блочных симметричных криптосистем

Цель работы - закрепление теоретических знаний и практическое освоение метода линейного криптоанализа блочных симметричных криптосистем на примере криптосистемы S-DES.

Описание лабораторной работы

Лабораторная работа выполняется с использование программы «Cryptoanaliz».

При подготовке к лабораторной работе студенты должны, используя литературу [1-4], материалы лекций углубить свои знания по следующим вопросам: блочные симметричные криптосистемы (определение, основные характеристики, достоинства и недостатки), блочная криптосистема S-DES, метод линейного криптоанализа блочных криптосистем, а также изучить инструкцию по использованию программы «Cryptoanaliz».

Преподаватель перед проведением занятия проводит контрольный опрос студентов и определяет степень их готовности к лабораторной работе.

Блочные симметричные криптосистемы

(БСК) представляют собой семейство обратимых криптографических преобразований блоков (частей фиксированной дины) исходного текста.

В настоящее время разработано большое количество БСК, многие из которых являются национальными стандартами. Наибольшую известность приобрели системы DES, IDEA, AES (Rijndael), ГОСТ 28147-89. Эти системы находятся под пристальным вниманием криптоаналитиков, основной задачей которых является поиск «слабых мест» в этих системах.

В настоящей работе метод линейного криптоанализа БСК рассматривается применительно к криптосистеме S-DES, являющейся упрощенной версией криптосистемы DES.

1. Алгоритм шифрования (расшифрования) криптосистемы S-DES. На рис. 1 иллюстрируется алгоритм шифрования (расшифрования).

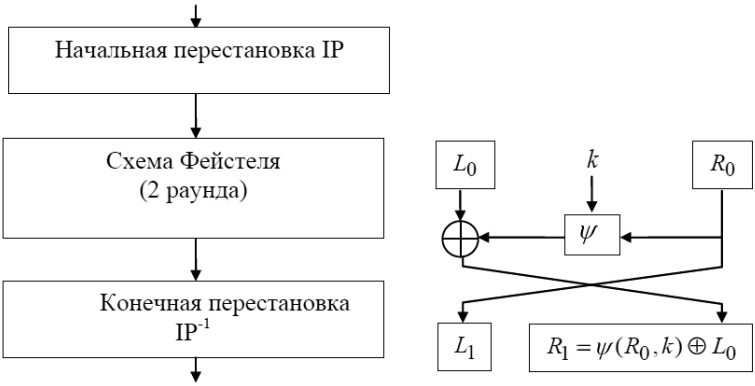

Рис. 1. Схема алгоритма шифрования –S-DES с сетью Фейстеля

Входной 8-битовый блок вначале подвергается начальной перестановке (IP), в соответствии с табл.1. Биты подблока пронумерованы от 0 до 7, причем бит с наибольшим порядковым номером 7 является младшим битом, и наоборот.

Таблица 1 – Начальная перестановка IP

7	6	4	0	2	5	1	3

Таблица разделена на две части, верхняя часть определяет подблок левых бит L_0 , а нижняя часть определяет подблок правых бит R_0 . Таким образом, после начальной перестановки IP, подблоки L_0 и R_0 подвергаются первому раунду шифрования. На выходе первого раунда получаются два выходных подблока L_1 и R_1 , полученные в соответствии с выражением:

$$L_1 = R_0;\ R_1 = L_0 \oplus \psi(R_0, k_{(8)_1}).$$

Функция ψ, называемая функцией усложнения и аналогичная функции усложнения алгоритма DES, зависит от ключа, а ее вид будет описан ниже.

Подблоки L_1 и R_1 являются входными для второго раунда шифрования, на выходе которого получаются подблоки L_2 и R_2 . Далее производиться объединение подблоков $L_2 \| R_2$ в блок, который подвергается перестановке, являющейся инверсией начальной перестановки. В результате получаем выходной блок криптограммы.

2. Алгоритм формирования раундовых ключей. Основной 10-битный ключ шифра $k_{(10)}$ используется для генерирования двух раундовых 8-битных

ключей $k_{(8)_1}$ и $k_{(8)_2}$. Основной ключ шифра $k_{(10)}$, биты которого пронумерованы от 0 до 9, подвергается перестановке РС-1, определяемой табл. 2.

Таблица 2 – Перестановка РС-1

9	7	3	8	0
2	6	5	1	4

Верхняя строка таблицы определяют биты (9,7,3,8,0) подблока C_0 , а нижняя - биты (2,6,5,1,4) подблока D_0 . Подблоки C_0 и D_0 подвергаются единичному сдвигу влево, результатом которого является подблоки C_1 и D_1.

Результат объединения подблоков $C_1 \| D_1$ подвергается перестановке, в соответствии с табл. 3.

Таблица 3 – Перестановка РС-2

5	3	9	7	2	8	6	4

Результатом перестановки РС-2 является первый раундовый ключ $k_{(8)_1}$.

Процедура формирования второго раундового ключа $k_{(8)_2}$ аналогична, отличие заключается в том, что подблоки C_1 и D_1 подвергаются двум сдвигам влево.

3. Функция усложнения. На рис. 2 представлена схема функции усложнения ψ.

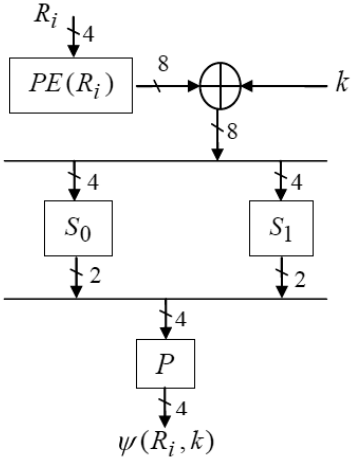

Рис. 2. Схемы функции усложнения

Вначале 4-х битный подблок подвергается перестановке с расширением (РЕ), в соответствии с табл. 4, на выходе которой получается 8-ми битный блок.

Полученный результат складывается по mod2 с битами 8-ми битного раундового ключа $k_{(8)\,i}$, i =1,2 и подвергается перестановке в блоках замены S_0 и S_1 (см. табл. 5 и табл. 6).

Таблица 4 – Перестановка с расширением РЕ

3	0	1	2	1	2	3	0

Таблица 5 – Блок замены S_0

S_0	№ столбца			
№ строки	**0**	**1**	**2**	**3**
0	1	0	2	3
1	3	1	0	2
2	2	0	3	1
3	1	3	2	0

Таблица 6 – Блок замены S_1

S_1	№ столбца			
№ строки	**0**	**1**	**2**	**3**
0	0	3	1	2
1	3	2	0	1
2	1	0	3	2
3	2	1	3	0

Причем, результат операции сложения по mod2 затем разбивается на два подблока, первые четыре бита (0,1,2,3) образуют подблок B_0 , оставшиеся биты (4,5,6,7) образуют подблок B_1 . Подблоки B_0 и B_1 подвергаются преобразованию в блоках замены S_0 и S_1 , соответственно. Крайние биты входного 4-битного подблока определяют строку таблицы, а средние биты – столбец. После преобразования в блоках замены выходные 2-битные подблоки объединяются S_0 (B_0) ‖ S_1 (B_1) . Полученный 4-битный подблок подвергается перестановке (Р), в соответствии с табл. 7.

Таблица 7 – Перестановка (Р)

1	0	3	2

Результатом перестановки будет выходное значение функции усложнения $\psi(R_i, k_{(8)_i}, i = 1,2)$

Метод линейного криптоанализа.

Метод линейного криптоанализа разработан в 1993 году японским криптологом Митсуру Матсуи. В первоначальном виде этот метод сформулирован применительно к криптосистеме DES, в настоящее время создаются новые модификации этого метода [4].

Идея метода линейного криптоанализа основана на том, что существует возможность заменить нелинейную функцию криптографического преобразования ее линейным аналогом. Линейный криптоанализ базируется на знании криптоаналитиком пар «открытый текст-криптограмма», а также алгоритма шифрования.

Будем считать, что при генерации исходного текста X случайные биты независимы и равновероятны $P(x\text{i} = 1) = \text{p}, P(x\text{i} = 0) = 1 - \text{p}, \text{p} = 0.5$ статистически аналогом (или приближенным линейным аналогом) называется выражение:

$$\lambda(X,Y) = \sum_{i=1}^{n} a_i x_i \oplus \sum_{i=1}^{n} b_i y_j = \sum_{k=1}^{L} c_k k_k\,, (1)$$

если вероятность

$$P\left\{\lambda, f(X,K) = \sum_{k=1}^{L} c_k k_k\right\} = 0.5 + \triangle.$$

Величина $\triangle = |1 - 2p|$ называется эффективностью линейного аналога, а коэффициент $a_i = \theta, 1, b_i = \theta, 1, c_k = \theta, 1$ - параметрами линейного аналога. По существу $\triangle$ характеризует степень линейности функции криптографического преобразования и имеет максимальное значение $\triangle_{max} = 0{,}5$. При применении метода линейного криптоанализа решаются две взаимосвязанные задачи:

1) нахождение эффективного линейного статистического аналога и вычисление его вероятности;

2) определение ключа (или нескольких бит ключа) с использованием эффективного линейного статистического аналога.

Практическая реализация метода линейного криптоанализа связана с реализацией следующих последовательных шагов.

1. Тщательно анализируется криптографическая функция и определяется множество линейных статистических аналогов. На этом шаге в первую очередь анализируются S – блоки функции усложнения ψ. Для этого формируются таблицы значений $Q_t(i,j)$, где: t=0,1 – номер S – блока, $i = \overline{1,4}, j = 1,2$.

Значение $Q_t(i,j)$ представляет собой количество совпадений суммы по mod2 некоторых битов входных данных. В ходе анализа прослеживаются все возможные комбинации двоичных векторов (j). Каждая пара векторов

используется в качестве маски, которая накладывается на возможные пары «вход – выход» S – блока. Эти маски указывают на биты входа и выхода, которые необходимо сложить по mod2, а затем сравнить полученные результаты. Далее проводится анализ полученных таблиц $Q_t(i,j)$ и отыскиваются такие значения i^*, j^*, для которых выполняется условие:

$$Q_t \sum_{i=1}^{n} a^*{}_i x_i \oplus : max|Q_t(i,j) - 8|. \quad (2)$$

В соответствие с полученной парой i^*, j^*, и учитывая в схеме алгоритма шифрования перестановки и сложения по mod2, формируется эффективный линейный статистический аналог:

$$\lambda^*(X,Y) = \sum_{i=1}^{n} a^*{}_i x_i \oplus \sum_{i=1}^{n} b^*{}_i y_j = \sum_{k=1}^{L} c^*{}_k k_k, P_{\text{эа}} = \frac{Q(i^*,j^*)}{16} \quad (3)$$

2. Генерируется множество независимых исходных текстов $\mathrm{X}^{(1)}$, $\mathrm{X}^{(2)}, \dots, \mathrm{X}^{(\mathrm{M})}$

и регистрируются соответствующие им криптограммы $Y^{(1)}$, $Y^{(2)}, \dots, Y^{(\mathrm{M})}$.

3. Для каждой пары $\mathrm{X}^{(m)}, Y^{(m)}, m = \overline{1,M}$ вычисляется значение левой части эффективного линейного статистического аналога:

$$\lambda^*\left(X^{(m)}, Y^{(M)}\right) = \sum_{i=1}^{n} a^*{}_i x_i^m \oplus \sum_{i=1}^{n} b^*{}_i y_i^m.$$

4. Определяется частота получения «1» при вычислении M значений (4):

$$\nu = \frac{1}{M} \sum_{m=1}^{M} \lambda^*\left(X^{(m)}, Y^{(M)}\right), \quad (5)$$

и строится оценка максимального правдоподобия в соответствии с правилом:

$$d = \begin{cases} 1, \nu \geq 0.5 \\ 0, \nu \leq 0.5 \end{cases} \quad (6)$$

5. Строится система линейных уравнений, причем каждое уравнение системы представляет собой равенство правой части (4) и соответствующего значения (6)

$$\sum_{k=1}^{L} c^*{}_k k_k = d. \quad (7)$$

Единственное решение полученной системы (7) используется в качестве оценки ключа $k^* = k_1{}^*, k_2{}^*, \dots, k_L{}^*$.

Задание

В процессе выполнения работы студенты должны:

1. Запустить на исполнение программу «Cryptoanaliz» и пройти предлагаемый контрольный тест.

2. Варианты индивидуальных заданий заложены в программе «Cryptoanaliz».В соответствии с заданием определенным преподавателем студенты выбирают номер варианта, количество известных текстов и осуществляют зашифрование случайным образом сгенерированных открытых текстов.

3. Используя таблицы Q_0 и Q_1, и учитывая таблицы перестановки и сложение по mod2, студенты определяют эффективные линейные аналоги и вычисляют их вероятности. Полученный результат студенты заносят в табл 8.

Таблица 8 – Эффективные линейные статистические аналоги

№ блока	Эффективный линейный аналог	p	$\Delta = \lvert 1 - 2p \rvert$
S_0			
S_1			

4. Для каждого из полученных линейных аналогов студенты определяют в соответствии с выражениями (5), (6) значение правой части уравнений используя модуль «Анализ».

5. Используя полученные результаты, студенты формируют систему уравнений (7). Решение системы уравнений позволяет определить все или часть битов 8-битных раундовых ключей. Используя алгоритм формирования раундовых ключей криптосистемы S-DES, студенты определяют основный 10-битный ключ шифра. Возможные варианты 10-битного ключа шифра и соответствующие ему 8-битные раундовые ключи студенты заносят в отчет по лабораторной работе.

6. Используя модуль «Проверка» студенты проверяют правильность каждого из полученных вариантов ключей шифра.

7. При совпадении результатов анализа с истинным ключом шифра студенты оформляют, в соответствии с требованиями настоящего пособия отчет и представляют его преподавателю для защиты.

8. Отчет должен включать в себя следующие пункты:

1. Схему блочной криптосистемы S-DES и исходные данные индивидуального задания.

2. Таблицы статистического анализа Q_0 и Q_1, и таблицу с эффективными линейными статистическими аналогами (табл. 8).

3. Систему линейных уравнений для определения битов ключа.

4. Варианты полученных ключей.

5. Результат проверки подтверждающий правильность определенного в работе ключа.

Контрольные вопросы

1. Блочные криптосистемы. Принципы построения. Достоинства и недостатки.

2. Режимы применения блочных криптосистем.
3. Схема Фейстеля.
4. Методы усложнения блочных шифров.
5. Криптосистема DES.
6. Криптосистема ГОСТ 28147 - 89.
7. Основная идея метода линейного криптоанализа.
8. Понятие эффективного линейного статистического аналога.
9. Методика применения линейного криптоанализа.

Литература

1. Баричев С.Г, Гончаров В.В., Серов Р.Е. Основы современной криптографии: Учебный курс. – 2-е изд. – М.: Горячая линия – Телеком, 2002.

2. Харин Ю.С., Беник В.И, Матвеев Г.В., Агиевич С.В. Математические и компьютерные основы криптологии: Учеб. пособие. – Минск: Новое знание, 2003.

3. Бабенко Л.К., Мишустина Е.А. Лабораторный практикум по изучению современных методов криптоанализа по курсу «Криптографические методы и средства обеспечения защиты информации». – Таганрог: Изд-во ТРТУ, 2003.

4. Бабенко Л.К., Мишустина Е.А. Изучение современных методов криптоанализа. Методическое пособие. – Таганрог: Изд-во ТРТУ, 2003

Лабораторная работа №5.

Изучение метода дифференциального (разностного) криптоанализа блочных симметричных криптосистем

Цель работы - закрепление теоретических знаний и практическое освоение метода дифференциального (разностного) криптоанализа блочных симметричных криптосистем на примере криптосистемы S-DES.

Описание лабораторной работы

Лабораторная работа выполняется с использование программы «Cryptoanaliz».

При подготовке к лабораторной работе студенты должны, используя литературу [1-4], материалы лекций углубить свои знания по следующим вопросам: блочные симметричные криптосистемы (определение, основные характеристики, достоинства и недостатки), блочная криптосистема S-DES, метод дифференциального (разностного) криптоанализа блочных криптосистем, а также изучить инструкцию по использованию программы «Cryptoanaliz».

Преподаватель перед проведением занятия проводит контрольный опрос студентов и определяет степень их готовности к лабораторной работе.

Метод дифференциального (разностного) криптоанализа предложен Э. Байхэмом и А. Шамиром, и, по мнению ряда специалистов компании IBM, является общим методом криптоанализа блочно-итерационных криптосистем.

Идея заключается в анализе процесса изменения несходства для пары открытых текстов $\Delta X = X \oplus X'$ имеющих определенные исходные различия, в процессе прохождения через циклы шифрования с одним и тем же ключом.

Пусть задана пара входов X и X', с несходством $\Delta X = X \oplus X'$. Известны перестановка IP и перестановка с расширением E, а следовательно, известны и несходства A на входе блоков замены S_0 и S_1. Выходы Y и Y' известны, следовательно, известно и несходство $\Delta Y = Y + Y'$, а значит, при известныхперестановках IP^{-1} и P известны несходства ΔC на выходе блоков замены S_0 и . S_1

Доказано, что для любого заданного $\triangle A$ не все значения ΔC равновероятны. Комбинация $\triangle A$ и ΔC позволяет предположить значения битов

для $E(X)\oplus K$ и $E(X')\oplus K$. То, что $E(X)$ и $E(X')$ известны, дает информацию о Ki .

Несходство различных пар открытых текстов приводит к несходству получаемых шифр-текстов с определенной вероятностью. Эти вероятности можно определить, построив таблицы для каждого из блоков замены. Таблицы сроятся по следующему принципу: по вертикали располагаются все возможные комбинации $\triangle A$, по горизонтали – все возможные комбинации ΔC, а на пересечении – число соответствий данного ΔC данному $\triangle A$.

Число наибольших совпадений указывает нам пару $\triangle A$ и ΔC, с помощью которой можно определить секретный ключ. Пара открытых текстов, соответствующих данным $\triangle A$ и ΔC называется правильной парой, а пара открытых текстов, не соответствующих данным $\triangle A$ и ΔC – неправильной парой.

Правильная пара подскажет правильный ключ цикла, а неправильная пара – случайный. Чтобы найти правильный ключ, необходимо просто собрать достаточное число предположений. Один из подключей будет встречаться чаще, чем все остальные. Фактически правильный подключ появляется из всех возможных случайных подключей.

Задание

В процессе выполнения работы студенты должны:

1. Запустить на исполнение программу «Cryptoanaliz» и пройти предлагаемый контрольный тест.

2. Варианты индивидуальных заданий заложены в программе «Cryptoanaliz». В соответствии с заданием определенным преподавателем студенты выбирают номер варианта и количество известных текстов.

3. Используя таблицы анализа несходств$(\triangle A, \Delta C)$ для блоков замены $S0$ и $S1$туденты определяют оптимальный дифференциал $(\triangle A^*, \triangle \mathrm{C}^*)$ и осуществляют зашифрование случайным образом сгенерированных открытых текстов. Программа выбирает из множества пар текстов пары удовлетворяющие оптимальному дифференциалу$(\triangle A^*, \triangle \mathrm{C}^*)$ и представляет их в виде в табл. 1.

Таблица 1 -Пары текстов удовлетворяющие оптимальному дифференциалу

N	X	$E(X)$	$S(E(X))$	Y
1				
…				
№	X'	$E(X')$	$S(E(X')$	Y'
1				
…				

4. Студенты анализируют пары открытых текстов и определяют множество раундовых ключей шифра и, соответственно, множество основных ключей шифра. Ключ, получаемый чаще остальных и будет наиболее вероятным ключом шифра.

5. Используя модуль «Проверка» студенты проверяют правильность определенного анализом ключей шифра.

6. При совпадении результатов анализа с истинным ключом шифра студенты оформляют, в соответствии с требованиями настоящего пособия отчет и представляют его преподавателю для защиты.

7. Отчет должен включать в себя следующие пункты:

1. Структуру алгоритма S-DES и таблицы перестановок и замен, соответствующие заданному варианту.

2. Результаты анализа таблиц замен и . 0 S1S

3. Результаты анализа пар открытых текстов.

4. Множество возможных раундовых ключей.

5. Результаты проверки, подтверждающие правильность определенного в работе ключа.

Контрольные вопросы

1. Основные понятия криптографии и криптоанализа.
2. Понятие блочной симметричной криптосистемы.
3. Основные характеристики блочных симметричных криптосистем.
4. Метод дифференциального (разностного) криптоанализа.

Литература

1. Баричев С.Г, Гончаров В.В., Серов Р.Е. Основы современной криптографии: Учебный курс. – 2-е изд. – М.: Горячая линия – Телеком, 2002.

2. Харин Ю.С., Беник В.И, Матвеев Г.В., Агиевич С.В. Математические и компьютерные основы криптологии: Учеб. пособие. – Минск: Новое знание, 2003.

3. Бабенко Л.К., Мишустина Е.А. Лабораторный практикум по изучению современных методов криптоанализа по курсу «Криптографические методы и средства обеспечения защиты информации». – Таганрог: Изд-во ТРТУ, 2003.

4. Бабенко Л.К., Мишустина Е.А. Изучение современных методов криптоанализа. Методическое пособие. – Таганрог: Изд-во ТРТУ, 2003