# Дисциплина «Защита информации»

# Лабораторная работа № 2

**Симметричное шифрование**

# Теоретическое введение

## История криптографии

История криптографии делится на 4 периода

- наивная (до 16 вв от Р.Х.) - большая часть шифров сводилась к перестановке и подстановке.

- формальная (16- нач.20 вв) - появление шифров стойких к ручному взлому. двухбуквенные шифры. квадрат Вижинера. Правило Керхгофса.

- научная (нач.20 в.- 60 гг. 20 в) - шифрование с помощью машин (Энигма и др.).

Теория Шеннона. Научное определение стойкого шифра с позиций теории информации .

- компьютерная ( 60-е гг. 20 в- настоящее время). Массовое использование ЭВМ для шифрования и расшифровывания информации. Первые открытые стандарт - DES(США-1974 г.), ГОСТ-89(СССР, 1989 г.).

Основное назначение криптографии - сохранить в тайне необходимую информацию. При этом не скрывается канал передачи информации и сам факт зашифровывания.

Правило Кергхофса - стойкость шифра должна определяться лишь секретностью ключа. То есть знание алгоритма шифрования НЕ должно давать никаких преимуществ злоумышленнику.

### Основные принципы блочного симметричного шифрования.

Блочные шифры обрабатывают кодируемое сообщение блоками из нескольких байт, при этом блок открытого текста X преобразуется в блок шифротекста Y того же размера с использованием некоторого ключа шифрования Key:

Y = Encrypt(X, Key)

Процедура дешифрации выполняет обратное преобразование, используя тот же самый ключ:

X = Decrypt(Y, Key)

В общем случае процедуры Encrypt и Decrypt не совпадают, однако если последовательность действий при шифрации и дешифрации в точности совпадает, блочный шифр называется *абсолютно симметричным*. Для абсолютно симметричного шифра, очевидно, справедливо:

X = Encrypt((Encrypt(X, Key), Key)

Преобразования Encrypt и Decrypt трактуют блоки открытого и зашифрованного текста как целые числа и выполняет над ними ряд арифметических либо логических действий, основная задача которых – тщательно «перемешать» биты блока открытого текста между собой, а также связать их с битами используемого ключа шифрования для формирования блока закрытого текста. Для того, чтобы все шифрующее преобразование было обратимым, действия, его составляющие должны быть также обратимы (обратимость действия означает, что по его результату и одному из операндов можно получить второй операнд). В таблице 2.1 приведен список обратимых операций, использующихся в современных криптографических преобразованиях [[6]](http://www.volpi.ru/education/zki/index.php?man=1&page=45).

Среди операций, приведенных в таблице, необходимо выделить операции умножения и табличной подстановки, которые относятся к классу нелинейных и существенно затрудняют методы линейного криптоанализа блочных шифров.

Таблица 2.1

|  |  |  |  |
| --- | --- | --- | --- |
| Основные обратимые операции | | | |
| **Название операции** | **Графическое обозначение** | **Формула преобразования** | **Обратное преобразование** |
| Сложение | http://www.volpi.ru/umkd/zki/images/imgTable-2.1.1.gif | X’ = X + V | Вычитание |
| Сложения по модулю 2 | http://www.volpi.ru/umkd/zki/images/imgTable-2.1.2.gif | X’ = X ⊕ V | Автообратима |
| Умножение по модулю 2N+1 (N - размер блока) | http://www.volpi.ru/umkd/zki/images/imgTable-2.1.3.gif | X’=(X·V) mod(2N+1) | Сомножитель можно найти по алгоритму Евклида |
| Циклические сдвиги вправо/влево | http://www.volpi.ru/umkd/zki/images/imgTable-2.1.4.gif | X’ = X ROR V X’ = X ROL V | Циклический сдвиг в обратном направлении |
| Табличная подстановка | http://www.volpi.ru/umkd/zki/images/imgTable-2.1.5.gif | X’ = SBox(X) | Обратная подстановка |

Для преобразований параметров криптоалгоритмов, которые не требуют обратимости, используются необратимые операции, такие как логические сложение и умножение, арифметические сдвиги влево и вправо, получение остатка от целочисленного деления, умножение по модулю 2N.

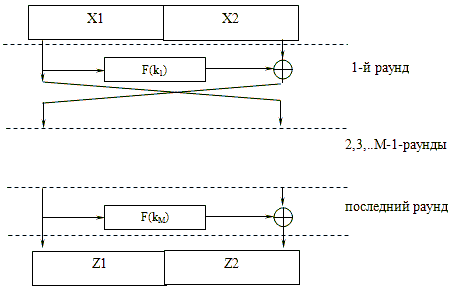
В качестве второго операнда V, участвующего в операциях криптографических преобразований, могут использоваться:

1. фиксированные числовые константы;
2. значения, вычисленные из независимой части шифруемого блока (например, можно сложить младшую и старшую часть блока шифруемой информации);
3. материал ключа – блок информации, вычисленный исключительно на основе информации, хранящейся в ключе шифрования.

Одним из основных принципов структурного построения современных криптоалгоритмов является принцип итерирования. Его идея заключается в многократной, состоящей из нескольких циклов (или раундов), обработки одного блока открытого текста с использованием на каждом циклеспециального ключа раунда, вычисляемого на основе ключа шифрования. Количество циклов можно варьировать из соображений криптостойкости и эффективности реализации алгоритма: увеличение количества циклов приводит к повышению стойкости шифра, но увеличивает время шифрации и потребляемые вычислительные ресурсы. Подобные циклические структуры принято называть *сетями*, и большинство современных блочных шифров построены с использованием сетевой архитектуры.

В качестве примера криптографических сетей можно привести SP-сети, содержащие в каждом раунде два слоя – подстановки (substitution), в котором обычно используются обратимые операции преобразования над шифруемым блоком и материалом ключа, и перестановки (permutation), в котором происходит перестановка бит внутри блока. Однако, самой популярной сегодня является сеть Фейштеля, схема которой представлена на рис.2.4.

При шифровании блок открытого текста разбивается на две равные части - левую и правую. На каждом цикле одна из частей подвергается преобразованию при помощи *образующей* функции F и вспомогательного ключа ki, полученного из исходного ключа. Результат преобразования складывается по модулю 2 с другой частью, после чего части меняются местами. Преобразования на каждом цикле идентичны и лишь после последнего раунда не выполняется перестановка частей блока.

  
Рис. 2.4. Схема сети Фейштеля на М раундов

Достоинством сети Фейштеля является то, что дешифрация выполняется той же последовательностью преобразований, лишь порядок следования вспомогательных ключей меняется на обратный. Это достигается за счет автообратимости операции суммы по модулю 2, смешивающей части блока между собой.

Если размер блока шифрования криптоалгоритма слишком велик, возможны модификации сети Фейштеля с 4 ветвями, один из вариантов которых приведен на рис. 2.5.

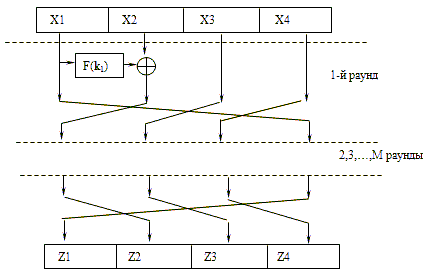
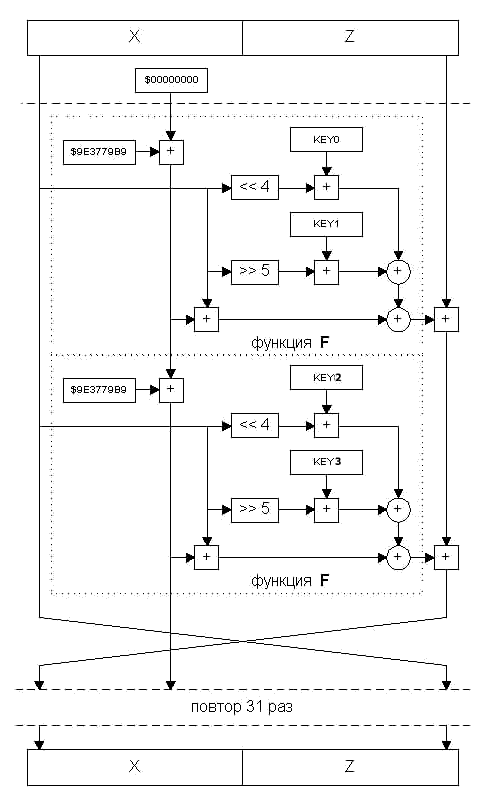
  
Рис. 2.5. Сеть Фейштеля с 4 ветвями

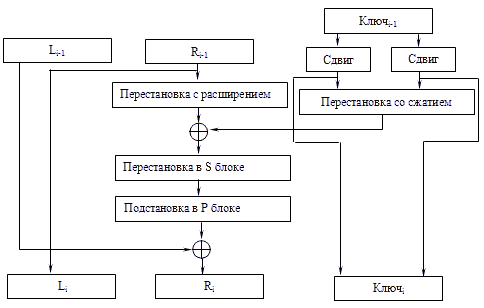
Схема Фейштеля используется в криптоалгоритмах DES, ГОСТ 28147-89, FEAL, Blowfish, LOKI, CAST и др. Рассмотрим практическую реализацию блочного шифра, построенного по принципу сети Фейштеля, на примере алгоритма TEA (Tiny Encrypton Algorithm).

Алгоритм TEA (рис.2.6) разработан в Кембриджском университете, представляет собой схему сети Фештеля на 64 раунда, размер шифруемого блока 64 бита, размер ключа шифрования 128 бит.

  
Рис. 2.6. Схема работы алгоритма TEA

Достоинством алгоритма TEA является простота реализации, однако,отсутствие в образующей функции нелинейных операций приходится компенсировать большим количеством раундов. КEY0, КEY1, КEY2, КEY3 на рис.2.6 – это ключи раунда, которые получаются простым делением ключа шифрования на 4 части. Необходимо также отметить, что TEA является несимметричным алгоритмом, поскольку ветви смешиваются обычным арифметическим сложением, поэтому операция дешифрации требует незначительных изменений.

Алгоритмы шифрования, получившие большее признание среди криптографов, используют в функциях преобразования нелинейные операции. Так, например, алгоритм шифрования DES (Data Encryption Standard), который до недавнего времени являлся мировым стандартом шифрования, использует в своей образующей функции операции табличной подстановки, которые существенно усложняют процедуру линейного криптоанализа этого шифра. Схема раунда алгоритма DES представлена на рис. 2.7.

  
Рис. 2.7. Раунд алгоритма DES

Каждый раунд преобразования включает перестановку правой части блока, причем на вход модуля расширения подается 32-разрядное число, а снимается с него 48-разрядное (некоторые биты входного числа копируются на выход дважды). После сложения с материалом ключа осуществляется табличная подстановка, в результате которой по 8 таблицам подстановки происходит замена 48-битного входа на 32-битный выход (каждая S-таблица подстановки заменяет 6-битный вход на 4-битный выход). Выход S-блока поступает на блок перестановки, где биты переставляются по законам, определяемым специальной P-таблицей. Заканчивается раунд традиционным для сетей Фейштеля смешиванием ветвей между собой. За счет использования нелинейных операций преобразований удалось уменьшить количество раундов до 16 при размере ключа 56 бит.

С момента опубликования алгоритма DES в 1977 году он подвергся серьезному криптоанализу. Так, в частности, были обнаружены 4 слабых ключа DES, которые половину из возможных 264 входных блоков кодируют самих в себя. Был предложен также ряд специфических атак на DES, которые еще в 80-е годы позволяли с использованием компьютеров специализированной архитектуры или распределенных вычислений провести успешную атаку на алгоритм в течение нескольких часов. Для современного технологического уровня время взлома шифра DES составляет несколько десятков минут. В связи с этим, для обеспечения более высокого криптостойкости рекомендуется применять тройное DES-шифрование на трех или двух различных ключах. Такие схемы называются EDE (encrypt-decrypt-encrypt) шифрованием. При двух ключах шифрования блок открытого текста сначала шифруется на ключе K1, затем дешифруется на ключе K2, а затем вновь шифруется на ключе K1. Размер ключевого пространства (общее количество возможных ключей шифрования) в этом случае возрастает до 2112 (у обычного DES – 256). При использовании трех ключей шифрования блок открытого текста сначала шифруется на ключе K1, затем дешифруется на ключе K2, а затемшифруется на ключе K3, что обеспечивает размер ключевого пространства 2168.

Из-за недостаточной длины ключа шифрования, а также ориентированности шифра DES на аппаратную реализацию (в связи с большим количеством используемых перестановок бит), американским национальным институтом стандартов в 1997 был объявлен конкурс на криптостандарт блочного шифрования AES (Advanced Encryption Standard). В 2000 году победителем конкурса был объявлен алгоритм Rijndael, разработанный бельгийскими криптографами. Его отличительной особенностью является то, что он построен не по принципу сети Фейштеля, а использует нетрадиционную структуру KALST- сети.

Алгоритм Rijndael представляет блок данных в виде двухмерного байтового массива размером 4х4, 4х6 или 4х8 (допускается использование нескольких фиксированных размеров шифруемого блока информации). Все операции выполняются с отдельными байтами массива, а также с независимыми столбцами и строками.

Алгоритм Rijndael выполняет четыре преобразования: BS (ByteSub) - табличная замена каждого байта массива, SR (ShiftRow) - сдвиг строк массива. При этой операции первая строка остается без изменений, а остальные циклически побайтно сдвигаются влево на фиксированное число байт, зависящее от размера массива. Например, для массива размером 4X4 строки 2, 3 и 4 сдвигаются соответственно на 1, 2 и 3 байта. Далее выполняется MC (MixColumn) - операция над независимыми столбцами массива , когда каждый столбец по определенному правилу умножается на фиксированную матрицу. И, наконец, AK (AddRoundKey) - добавление ключа. Каждый бит массива складывается по модулю 2 с соответствующим битом ключа раунда, который, в свою очередь, определенным образом вычисляется из ключа шифрования. Количество раундов шифрования в алгоритме Rijndael переменное (10, 12 или 14 раундов) и зависит от размеров блока и ключа шифрования (для ключа также предусмотрено несколько фиксированных размеров).

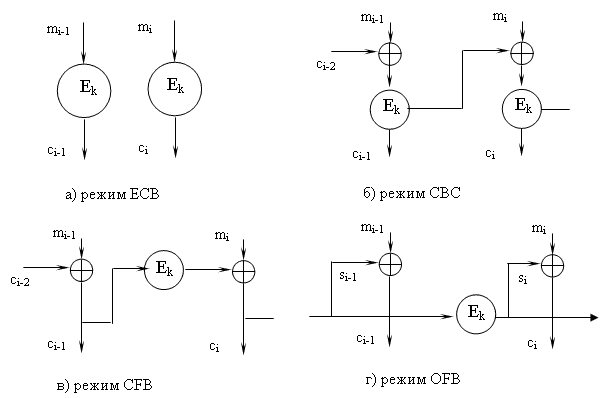
Достоинствами шифра Rijndael является то, что он обеспечивает высокую скорость шифрования на всех платформах: как при программной, так и при аппаратной реализации. Его отличают несравнимо лучшие возможности распараллеливания вычислений по сравнению с другими алгоритмами, представленными на конкурс AES. Кроме того, требования к ресурсам для его работы минимальны, что важно при его использовании в устройствах, обладающих ограниченными вычислительными возможностями.

Российским стандартом блочного шифрования является алгоритм ГОСТ 28147-89, построенный по структуре сети Фейштеля. Его особенностью следует признать большой размер ключа (256 бит), непубликуемые таблицы подстановки, большое количество раундов (32 раунда).

### Режимы шифрования блочных шифров.

Применение блочного шифра для нужд практического кодирования информации возможно в одном из четырех основных режимов шифрования (рис. 2.8):

1. режим электронной кодовой книги (Electronic Code Book, ECB);
2. режим сцепления блоков шифра (Cipher Block Changing, CBC);
3. режим обратной связи по шифротексту (Electronic Feedback, CFB);
4. режим обратной связи по выходу (Output Feedback, OFB).

  
Рис. 2.8. Режимы шифрования блочных шифров

В режиме ECB шифрование/дешифрование i-го блока открытого текста/шифротекста выполняется независимо от остальных блоков:

ci = Ek(mi), mi = Dk(ci).

Недостатком данного режима шифрования является то, что одинаковые блоки входного текста будут кодироваться в одинаковые блоки шифротекста, что дает возможность злоумышленнику, во-первых, делать предположения о характере информации в открытом тексте, а во-вторых, подменить один или несколько блоков шифротекста. Достоинством режима можно назвать простоту реализации, а также возможность распараллеливания процедуры шифрации.

Режим CBC предполагает следующие алгоритмы шифрации/дешифрации:

ci = Ek(mi ⊕ ci-1),mi = Dk(ci) ⊕ ci-1

В режиме CBC каждый блок открытого текста складывается с блоком шифротекста, полученным на предыдущем этапе. Таким образом, происходит сцепление блоков друг с другом и независимая манипуляция с каждым из них невозможна, а одинаковые входные блоки будут давать на выходе разные блоки. Однако, задача распараллеливания процедуры кодирования в этом режиме затруднена. Дополнительным параметром процедур шифрования/дешифрования является параметр c0.

В режиме CFB также происходит «маскировка» блока открытого текста уже зашифованными блоками:

ci = mi ⊕ Ek(ci-1), mi = Dk(ci-1) ⊕ ci

По своим возможностям данный режим похож на режим CBC, но если длина сообщения не кратна размеру блока шифра, то в режиме CBC необходимо дополнять последний блок дополнительными битами и сообщать на принимающую сторону истинный размер сообщения, а режим CFB позволяет сформировать шифротекст того же размера, что и исходное сообщение.

В режиме OFB исходное сообщение вообще не подвергается криптопреобразованию, оно складывается с шифруемыми на секретном ключе блоками si (s0 является задаваемым несекретным параметром режима):

ci = mi ⊕ si, mi = ci ⊕ si, si = Ek(si-1)

В этом режиме, как и в режиме ECB, ошибки, которые могут возникнуть при передаче шифротекста по каналам связи, локализуются в блоке, не распространяясь на соседние, причем в режиме OFB ошибочными будут только биты, подвергшиеся изменению (в ECB изменится весь блок). Это дает возможность злоумышленнику незаметно для принимающей стороны подменить блок шифротекста. Возможности распараллеливания процедур шифрации/дешифрации затруднены.

# Практические задания

Написать програму которая шифрует или расшифровывает текстовой файл длины не менее 32 байт. Максимальный размер == 4 Мб.

Программа должна работать в 2-х режимах: шифрования и расшифрования.

Режим шифрование :

Входные параметры : имя исходного , пароль

Выходные параметры : имя зашифрованного файла

Режим расшифрования :

Входные параметры : имя исходного файла , пароль .

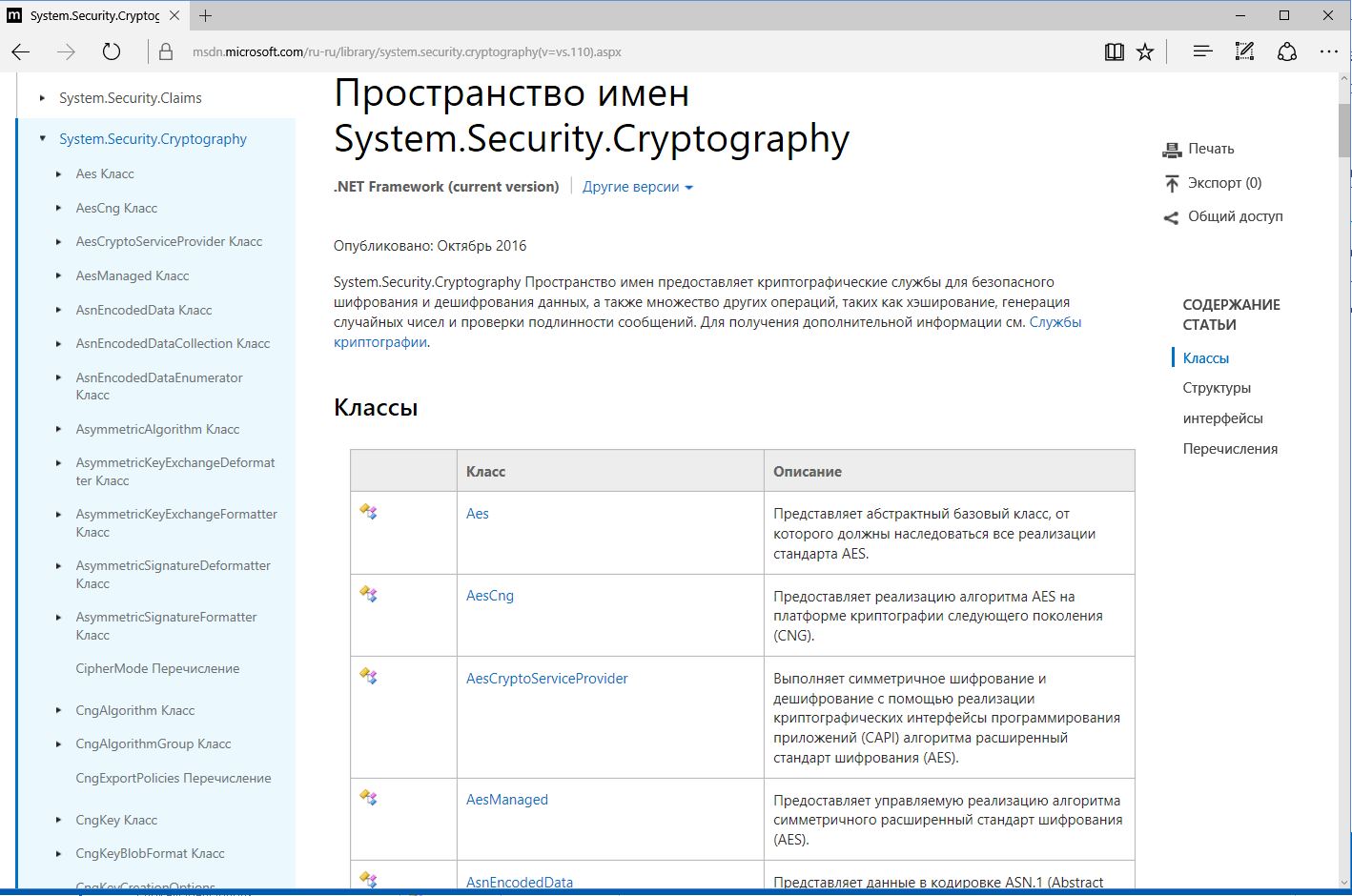
Выходные параметры : имя расшифрованного файла

|  |  |  |
| --- | --- | --- |
| Вариант | Алгоритм | Примечания |
| 1 | DESX |  |
| 2 | 3DES | Надо задать три ключа |
| 3 | ГОСТ-89 |  |
| 4 | AES-128 |  |
| 5 | AES-192 |  |
| 6 | AES-256 |  |

Примечания : внутри программы должен использоваься ключ . Пароль не должен храниться внутри программы в исходном виде (иначе его легко можно определить простым просмотром в 16-ном редакторе). Шифрование пароля с помощью побитовой операции XOR также не допускается. Если пароль вводится в графическом режиме, то он должен заменяться символами «\*» (звездочка).

### Пример выполнения практической части

Программу на C#, реализующую алгоритм AES можно реализовать с использованием пространства имён System.Security.Cryptography.



Описание пространства имён System.Security.Cryptography

# Контрольные вопросы

1. Определение симметричной криптографии.

2. Его достоинства и недостатки сим. криптографии.

3. Краткая характеристика алгоримов DES,3DES, ГОСТ89, AES.

Допольнительные.

4. Принципы Керхгоффса.

5. Работа Шэннона ( кр. характеристика) "О секретных системах).

6. Сеть Фейстеля и какое отношение к ней имеет работа Шэннона.

**Литература.**

1. Алферов, Зубов, Кузьмин, Черемушкин. Основы криптографии.

2.Шнаер Б. Прикладная криптография.

3. Жельников В. Криптография от папируса до компьютера.

4. Аграновский , Хади. Практическая криптография.

5. Лясин Д. Н., Саньков С. Г. Методы и средства защиты компьютерной информации