Расчётно-графическая работа

Новый стандарт симметричного шифрования AES

Цель работы

Знакомство с новым стандартом симметричного шифрования AES (Rijndael), принятым в 2001 году. Приобретение практических навыков использования данного алгоритма шифрования.

Описание алгоритма AES

AES (Rijndael) представляет собой 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 (стандарт AES использует длину блока только 128 бит) [1, 2]. Алгоритм шифра представлен на рисунке 1:

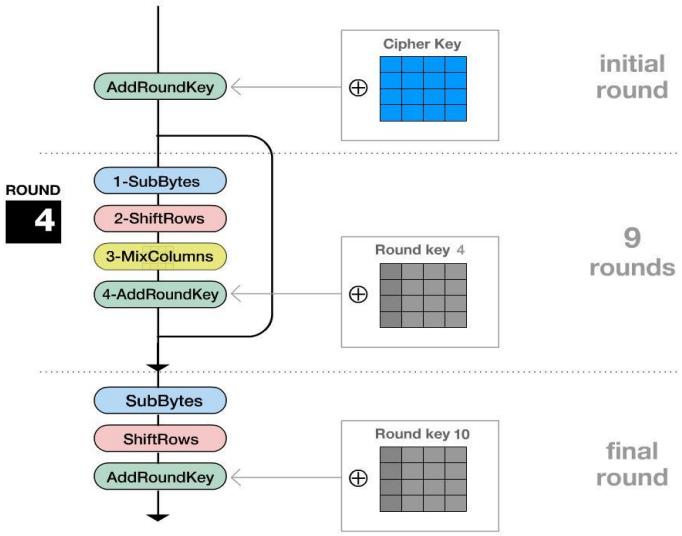

Рисунок 1 – Процесс шифрования текста алгоритмом AES

Состояние, ключ шифрования и число циклов

Разнообразные преобразования работают с промежуточным результатом шифрования, называемым ***состоянием*** (State).

Состояние можно представить в виде прямоугольного массива байт. Этот массив имеет 4 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. Это показано на рисунке 2:

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

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

Рисунок 2 – Пример представления состояния ($Nb = 6$) и ключа шифрования ($Nk = 4$)

В некоторых случаях ключ шифрования показан как линейный массив 4-байтных слов. Слова состоят из 4 байт, которые находятся в одном столбце (при представлении в виде прямоугольного массива).

Входные данные для шифра (“открытый текст”) обозначаются как байты состояния в порядке $a_{0,0}$, $a_{1,0}$, $a_{2,0}$, $a_{3,0}$, $a_{0,1}$, $a_{1,1}$, $a_{2,1}$, $a_{3,1}$, $a_{0,2}$, … После завершения действия шифра выходные

данные получаются из байт состояния в том же порядке. Число циклов обозначено как Nr и зависит от значений Nb и Nk. Оно приведено в таблице 1.

Таблица 1 – Число циклов (Nr) как функция от длины ключа и длины блока

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 псевдокода это выглядит следующим образом:

```
Round (State, RoundKey)
{
 SubBytes (State);                // замена байт
 ShiftRows (State);               // сдвиг строк
 MixColumns (State);              // замешивание столбцов
 AddRoundKey (State, RoundKey);   // добавление циклового ключа
}
```

Последний цикл шифра немного отличается:

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

В приведённой записи функции **`Round ()`**, **`SubBytes ()`** и другие выполняют свои действия над массивами, указатели на которые (т.е. **`State`**, **`RoundKey`**) им передаются.

Как можно заметить, последний цикл отличается от простого цикла только отсутствием замешивания столбцов. Каждое из приведённых преобразований разобрано далее.

Замена байт (SubBytes)

Блоки замен в шифре Rijndael играют важную роль. Согласно основополагающим принципам, сформулированным ещё К. Шенноном, преобразования данных, используемые в шифре, должны придавать последнему два основных свойства – рассеивание и перемешивание.

Преобразование **`SubBytes`** представляет собой нелинейную замену байт, выполняемую независимо с каждым байтом состояния. Таблицы замены (или ***S-блоки***) являются инвертируемыми и построены из композиции двух преобразований:

1. Получение обратного элемента относительно умножения в поле $GF(2^8)$, при этом $'00'$ переходит сам в себя.
2. Применение аффинного преобразования (над $GF(2^8)$), определённого как:

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}.$$

Применение описанного *S*-блока ко всем байтам состояния обозначено как **`SubBytes (State)`**. Рисунок 3 иллюстрирует применение преобразования **`SubBytes`** к состоянию.

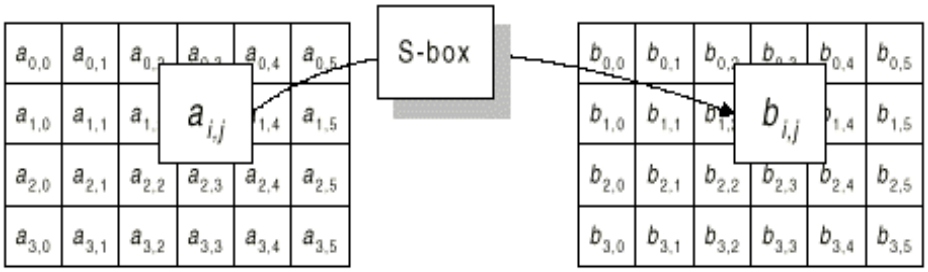

Рисунок 3 – SubBytes действует на каждый байт состояния

Преобразование сдвига строк (ShiftRows)

Последние 3 строки состояния циклически сдвигаются на различное число байт. Строка 1 сдвигается на C_1 байт, строка 2 – на C_2 байт и строка 3 – на C_3 байт. Значения сдвигов C_1, C_2 и C_3 зависят от длины блока Nb. Их величины приведены в таблице 2.

Таблица 2 – Величина сдвига для разной длины блоков

Nb	C_1	C_2	C_3
4	1	2	3
6	1	2	3
8	1	3	4

Операция сдвига последних 3 строк состояния на определённую величину обозначена как **`ShiftRows (State)`**. Рисунок 4 показывает влияние преобразования на состояние:

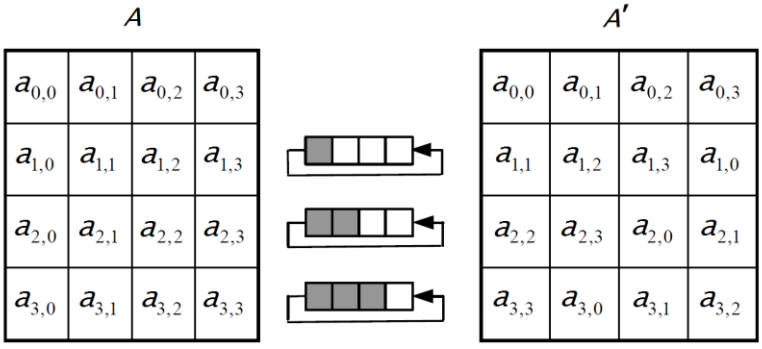

Рисунок 4 – ShiftRows действует на строки состояния

Преобразование замешивания столбцов (MixColumns)

В этом преобразовании столбцы состояния рассматриваются как многочлены над $GF(2^8)$ и умножаются по модулю $x^4 + 1$ на многочлен $c(x)$, имеющий следующий вид:

$$c(x) = '03' \cdot x^3 + '01' \cdot x^2 + '01' \cdot x + '02'.$$

Это может быть представлено в виде матричного умножения. Пусть $b(x) = c(x) \cdot a(x)$, тогда

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \cdot \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

Применение этой операции ко всем четырём столбцам состояния обозначено как **`MixColumns (State)`**. Рисунок 5 демонстрирует применение **`MixColumns`** к состоянию:

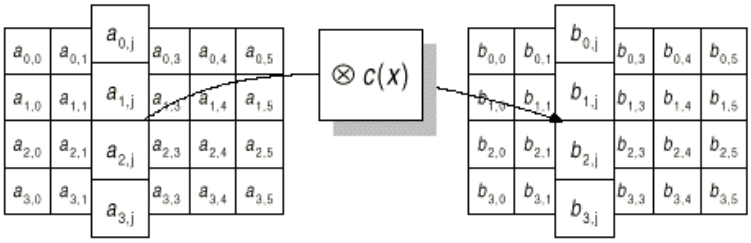

Рисунок 5 – MixColumns действует на столбцы состояния

Добавление циклового ключа

В данной операции цикловой ключ добавляется к состоянию посредством операции XOR. Цикловой ключ вырабатывается из ключа шифрования посредством алгоритма выработки ключей (Key Schedule). Длина циклового ключа равна длине блока Nb. Преобразование, содержащее добавление посредством XOR циклового ключа к состоянию, обозначено как **`AddRoundKey (State, RoundKey)`** и проиллюстрировано на рисунке 6:

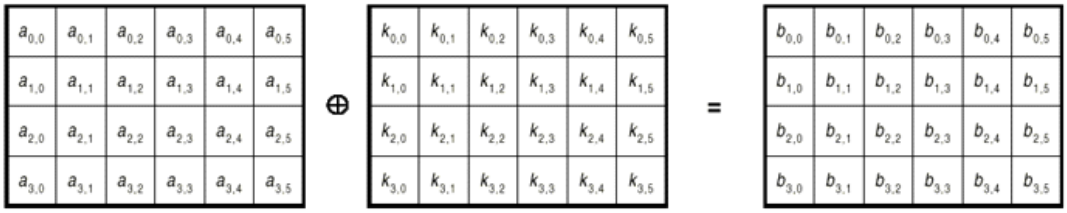

Рисунок 6 – Формирование циклового ключа

Алгоритм выработки ключей (Key Schedule)

Цикловые ключи получаются из ключа шифрования посредством алгоритма выработки ключей. Он содержит два компонента: расширение ключа (Key Expansion) и выбор циклового ключа (Round Key Selection). Основополагающие принципы алгоритма выглядят следующим образом:

1. Общее число бит цикловых ключей равно длине блока, умноженной на число циклов плюс 1 (например, для длины блока 128 бит и 10 циклов требуется 1408 бит циклового ключа).
2. Ключ шифрования расширяется в расширенный ключ (Expanded Key).
3. Цикловые ключи берутся из расширенного ключа следующим образом: первый цикловой ключ содержит первые Nb слов, второй – следующие Nb слов, и т.д.

Расширение ключа (Key Expansion)

Расширенный ключ представляет собой линейный массив четырёхбайтных слов и обозначен как $W[Nb \cdot (Nr + 1)]$. 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 $Nk \leq 6$ и версия для $Nk > 6$.

Для $Nk \leq 6$ имеем:

```
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] ^ SubBytes (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];
    }
}
```

Как можно заметить, первые Nk слов заполняются ключом шифрования. Каждое последующее слово $W[i]$ получается посредством применения операции XOR для предыдущего слова $W[i-1]$ и слова на Nk позиций ранее $W[i-Nk]$. Для слов, позиция которых кратна Nk, перед операцией XOR применяется преобразование к $W[i-1]$, а затем ещё прибавляется цикловая кон-

станта. Преобразование содержит циклический сдвиг байт в слове, обозначенный как **Rotl**, а затем следует **SubBytes** – применение замены байт.

Для $Nk > 6$ имеем:

```
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] ^ SubBytes (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] ^ SubBytes (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 **SubBytes** для каждого 4-го 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]);
```

Выбор циклового ключа

i-й цикловой ключ получается из слов массива циклового ключа от $W[Nb \cdot i]$ и до $W[Nb \cdot (i+1)]$. Это показано на рисунке 7:

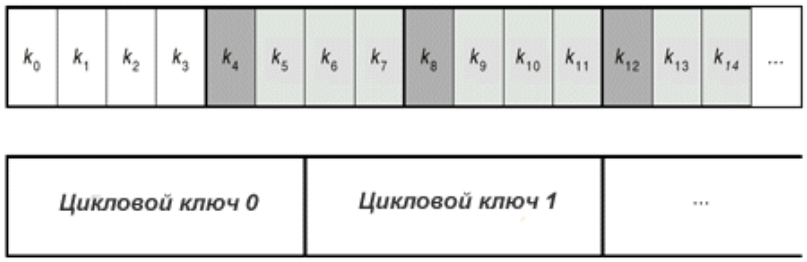

Рисунок 7 – Расширение ключа и выбор циклового ключа для $Nb = 6$ и $Nk = 4$.

Замечание: Алгоритм выработки ключей можно осуществлять и без использования массива $W[Nb \cdot (Nr+1)]$. Для реализаций, в которых существенно требование к занимаемой памяти, цикловые ключи могут вычисляться на лету посредством использования буфера из Nk слов.

Общий алгоритм шифрования

Шифр Rijndael состоит из:

- начального добавления циклового ключа;
- $Nr - 1$ циклов;
- заключительного цикла.

На языке псевдокода это 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 Rijndael 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);
}
```

Замечание: Расширенный ключ **всегда** должен получаться из ключа шифрования и никогда не указывается напрямую. Нет никаких ограничений на выбор ключа шифрования.

Задание

Для заданных в варианте открытого текста и ключа выполнить первый раунд шифрования по алгоритму AES.

Требования к оформлению отчёта

В отчёте в шестнадцатеричном виде должны быть приведены значения, находящиеся в состоянии (**`State`**) после:

- сложения открытого текста с раундовым ключом перед первым раундом;
- преобразования **`SubBytes`**;
- преобразования **`ShiftRows`**;
- преобразования **`MixColumns`**;
- сложения текущего состояния с раундовым ключом после первого раунда (это и есть главный результат РГР).

В отчёте должны быть ***подробно*** приведены все вычисления.

Критерии оценивания качества работы

1. Правильность ответов:
   1 – все требуемые значения найдены правильно;
   0 – не более 4 байт главного результата содержат ошибки;
   л.р. не принимается – более 4 байт главного результата содержат ошибки.
2. Подробность решения:
   1 – все расчёты приведены подробно;
   0 – не все расчёты приведены подробно.
3. Глубина понимания материала расчётно-графической работы:
   1 – быстрые и правильные ответы на все вопросы;
   0 – не на все вопросы ответы правильные и быстрые;
   л.р. не принимается – на половину вопросов ответы неправильные.

Варианты

Вар.		Открытый текст и ключ
1	P=	5D 3F 3F 7A 2D 3F 06 3F AC 2C 64 26 2B 1B 60 11
	K=	3F 3F 5A 76 6F 12 2B 38 3F 53 19 5A 5D 3C A6 41
2	P=	32 7B BB 08 6A AB 3F 65 4D AC 45 75 A6 4D 6E 27
	K=	5C 47 2B 2B 3F 3F 6F 33 35 69 13 AC 3F 5A 61 31
3	P=	2D 2B 65 65 A6 2D A6 3F 3F 48 74 3F 65 49 55 29
	K=	3F 04 3F 75 6F 0A 41 4C 09 B7 0C 3F 3F 34 4F 3F
4	P=	48 69 71 5A 12 3F 74 AC 6D 28 11 A6 34 51 2B 6F
	K=	95 2D 65 69 2D 53 7D AC 64 A6 24 6E 0A 28 42 57
5	P=	45 62 3F 35 3D 2F 1C 54 03 61 6E 3F 6B 0B 4E 70
	K=	4B 04 2D 75 28 3A A6 3F 6B 0B 1E 61 20 28 74 79

Вар.		Открытый текст и ключ
6	P=	50 60 0B 17 3F 2B 43 3A 43 3F 79 54 41 2B 65 24
	K=	2D 54 3F A6 3F 54 1F 36 43 B1 AC 3F 42 69 2D 65
7	P=	5A 23 0F 55 64 76 79 65 AC 4E 75 3F 49 75 B1 61
	K=	3F 09 35 03 62 45 7D 61 49 2B 4E 2D 6E 5E 2B 10
8	P=	3F 33 61 3F 3F 2B 3F 47 69 4A 31 2D 2B 95 21 3F
	K=	75 4C 65 69 76 40 2B A6 2B 2D AB 7F 6D A6 09 3F
9	P=	A6 4E 58 B1 2D 69 25 A6 5F 65 5C 3F 2D 65 61 2C
	K=	09 1E 41 6D 4F 3F B1 49 15 4E 31 3F A6 74 0F 38
10	P=	45 3F 6F 2A 6F 65 05 A6 03 56 3F AC 3F 04 5C 2E
	K=	3F 2B 6F 3F 2D 31 2D 72 15 E3 2A 05 3F 69 3D 62

Вар.		Открытый текст и ключ
11	P=	3F A6 3F 3F 3F 41 3F 43 75 79 1D 22 42 3F 3C 4C
	K=	20 55 3F 44 75 AC 28 A6 6B 2D 27 3F 76 32 4E 6F
12	P=	12 3F 33 3F 22 76 74 4E 65 34 64 75 57 75 25 2D
	K=	3F 45 2D 3F 2D AB 2B 3D 50 61 4C 30 65 29 2E 4E
13	P=	75 75 B1 79 58 4F 1A 3F 3F A6 53 AC 23 48 5F 7F
	K=	64 7E 71 AC 51 3F 26 58 6F 10 11 2D 6A A6 2D B7
14	P=	3F 2B 3F 56 A0 2D A6 69 3F AC 11 3F 54 4F 5E 79
	K=	3C A6 3F 0E 0C B7 2B 4F 43 4C 63 45 2D 53 3F 75
15	P=	61 30 45 54 3F 29 3F 35 3F 2C 37 3F 55 21 26 AC
	K=	2D 2B AB 2B 4E 5A 6F 2C A6 63 6F 49 2D 25 69 3F
16	P=	3F 40 18 4E 2D 3F 42 79 65 AC 6F 59 2D 3F 2D 54
	K=	A6 3F 6A 79 76 BB 6E 76 61 0A 6F 09 2F 3F 65 2D
17	P=	3F 38 3F 1C 66 68 95 10 28 A6 11 64 AC 3D 2D 36
	K=	75 7F 3F 39 3F 3F 24 6C 41 3F 10 2D 2D 58 62 3F
18	P=	37 0F 3F 3F 26 17 37 42 54 3F 3F 2B 3F 3E 3F 01
	K=	14 41 A6 6B 59 66 60 69 3C 69 3F 68 0B 2E 3F 13
19	P=	3F 7A 3F E3 A6 3E 2D 32 19 6F 2B 72 2C B0 6C 2B
	K=	3F 09 34 4F 20 7F 5F 61 75 3F 7C 52 5C 3F 65 04
20	P=	12 75 2D 1B 0C 30 67 0E 14 A6 04 A6 1C 54 59 54
	K=	69 4E B7 37 29 54 AC 61 3F 4F 14 3F A6 2D 28 48
21	P=	35 61 60 A6 61 13 A6 A6 55 41 3F 3F 69 49 6F 3F
	K=	31 3F 43 B0 A6 A6 69 2D 3F 3F 2D 6F 5A 69 2D 2B
22	P=	7E 5C B7 60 18 4C 4C 62 6E 4F 13 A6 2D 3F 14 66
	K=	2D 6B 7C 09 54 02 A6 45 69 54 23 50 2D 4C 3F 24
23	P=	26 0E 3F 2D 6F 2D 65 25 2D 65 3A 03 3D 16 75 39
	K=	2D A6 61 2B 3F 0B A6 3F 4C 61 27 3F B1 AC 3F 3F
24	P=	B1 3F 5B 30 75 19 7B B0 49 3E 56 17 7C 68 3D 45
	K=	3F 33 BB 72 3F 69 3F 3D 3F 14 53 0A 71 3F 10 13
25	P=	28 3F 2B 2D 6F 47 4B 05 4E 3F 76 75 54 6F 61 3F
	K=	3F 24 2D AC 2B 79 49 5B 65 6F 3F 3F 2C 4F 25 3F
26	P=	25 54 3F 3F 2D 3F 2C 3F 54 63 2B 54 2B 2D 3F 2D
	K=	3F 2D 4B 95 4A 5D 2D 07 54 40 4F 16 21 2D 2D 5F
27	P=	40 3F B1 63 61 6B 3A 1B 3F 3F 7C 54 3F 3F 33 A6
	K=	4D 75 61 BB 65 72 2D 63 7B 11 4A 63 61 56 08 00
28	P=	44 3F 73 3F 2B 2B A6 A6 0F 48 0A 49 37 78 54 3F
	K=	33 3F 34 75 2C 50 17 3F 5F 3F 6B 55 6F 45 3F A6
29	P=	14 3F 31 69 2D 67 6A 61 2D 75 2B 14 2D 3C 5A 3F
	K=	3F A6 3C 12 12 1F 76 67 3F 2D 3F 41 3F 31 A6 14
30	P=	1C 65 3F 55 AC 2F 4E 5B 75 51 52 37 46 3F 3F B7
	K=	AC 55 14 3C 76 6E 61 2B 43 3F 56 A6 AC 54 2D 2B

Вар.		Открытый текст и ключ
31	P=	18 3F 77 44 A6 60 3F 42 58 6D 3F 75 74 6B 39 42
	K=	50 2D 02 0E 3F 72 49 49 01 13 3A 65 4F 49 71 72
32	P=	5B B0 17 41 75 A6 49 2D 7D 65 75 3F 23 71 2D 04
	K=	21 0F 75 2D 24 19 B7 2D 14 A6 AC 65 0E 2B 64 3F
33	P=	3C 6F 58 0C 2B 19 1D 79 3F 10 3F 35 75 A0 79 22
	K=	4C 0C 4C 53 44 3D 19 A6 07 E3 51 67 4E 04 5E 65
34	P=	65 6F 3F A6 4B 2A 22 3F 67 6F 2D 02 6F 4D 2D 51
	K=	61 3F 4B 3F 61 3F 10 5E 3F 2E 95 69 54 3F 45 71
35	P=	3F 6D 75 27 95 67 71 48 2B 7E 3F 41 AC 3F 03 3F
	K=	A6 7B 06 1D 24 1B 60 14 4D 43 2F 2B 2B 2B 44 AB
36	P=	1C 5A 6F 3F 3B 2D 3F 2D 2D 61 75 45 79 59 06 2A
	K=	6F 54 2D 3F 10 75 49 01 75 61 6F 3F 2D 2B 4C 3F
37	P=	4C 39 3F 2D 65 A6 75 2D 32 1A 75 29 A6 57 23 3F
	K=	61 2D 02 7B 67 75 20 4C 2E 4F B0 10 7C 69 65 41
38	P=	56 2A 54 6A 7F 2D 6F 3F 09 43 31 19 4B 3E 44 66
	K=	A6 3F 08 A6 3F 2D 2D 25 78 A6 61 01 07 27 5E 3F
39	P=	41 68 55 3F 7F 64 08 2B 3F 36 2B 3F 0E 2B 45 64
	K=	40 3F 2D A6 09 1E 01 62 69 4C 4C 30 3E 6F 3B 41
40	P=	36 61 AC 2D 37 5E 55 03 3F 5A 61 2B 2D 5C 5C 6C
	K=	2F 33 41 1E 3F 37 3C 3B 13 5C AC 25 69 61 2A 3F
41	P=	03 A6 A6 76 72 3F 74 65 33 62 50 69 41 10 10 46
	K=	3F A6 AC 3F 76 4C 18 1B 3F A6 10 7E 3F 31 3F 13
42	P=	3E 14 22 41 3F 22 41 4C 2D 6E 7F 2D 40 18 48 4C
	K=	E3 B0 3F 04 BB AC AC 3F 4E 6F 3F 2D 3F A6 2B A6
43	P=	50 6E 3F 3F 1B 03 2B 3F 63 65 E3 A6 10 6C 26 45
	K=	6F 5B 1F AB 05 54 E3 3F 54 2D 2B 2D 17 74 69 75
44	P=	41 41 1A 3F 50 3F 79 3F 3F E3 6F 3F 56 6F 3F 45
	K=	55 4C 3F 54 26 64 3F 7D 78 69 2D 5E A0 04 3F 3F
45	P=	34 20 A6 51 6F 10 19 59 29 38 2D A6 13 75 A6 21
	K=	41 14 3F 3F 77 3F 7F 63 61 65 3F 3F 3F 2A A6 3F
46	P=	2D 42 49 7D 69 AC 54 45 A6 36 2C 3F 61 31 4E 2D
	K=	3F 4A 26 64 6E 1D 31 1D 4C 1E 6E 2B 71 16 A6 3F
47	P=	A6 A6 53 3F 76 A6 06 3F 50 42 17 10 08 2D 95 65
	K=	4B 2D A6 A6 3F 18 6C 54 61 71 36 69 A6 A6 10 3F
48	P=	61 36 33 1E 17 69 3F 75 3F 45 2D 2D 50 10 4D 3F
	K=	02 AB 55 64 54 60 61 3F 7E 0A 6F 7F 69 40 75 63
49	P=	74 3F 1B 62 3F 18 AC 7B 75 7D 76 65 75 3F B7 3B
	K=	2D 3F 1E 76 3F 02 3F B0 66 03 7E 6F AC AC 5B 32
50	P=	78 73 3F A0 17 3B 41 55 2D 3F 3E 61 3F A6 26 61
	K=	79 3F 58 A6 3F A6 2D 3F 2B 75 4D 3F 6F 14 2F 78

Вопросы для защиты

I. **Первая часть защиты** (обязательная):
   1. В чём заключается преобразование **`SubBytes`**?
   2. В чём заключается преобразование **`ShiftRows`**?
   3. В чём заключается преобразование **`MixColumns`**?
   4. В чём заключается преобразование **`AddRoundKey`**?
   5. В чём заключается расширение ключей (Key Expansion)?

II. **Вторая часть защиты**: Найти, чему равно $(a \cdot b)_{16}$ по модулю многочлена

$$m(x) = x^8 + x^4 + x^3 + x + 1, \text{ если:}$$

1. $a = F9_{16}, \quad b = 9C_{16}.$
2. $a = 63_{16}, \quad b = BD_{16}.$
3. $a = 3E_{16}, \quad b = EB_{16}.$
4. $a = 84_{16}, \quad b = 6E_{16}.$
5. $a = D9_{16}, \quad b = 91_{16}.$
6. $a = ED_{16}, \quad b = FB_{16}.$
7. $a = 96_{16}, \quad b = B1_{16}.$
8. $a = 5D_{16}, \quad b = 21_{16}.$

Список литературы

1. Мао, В. Современная криптография: теория и практика : Пер. с англ. / В. Мао. – М. : Издательский дом "Вильямс", 2005. – 768 с.
2. Advanced Encryption Standard (AES): FIPS Publication 197. – Springfield : National Technical Information Service, 2001. – 47 p.