МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №5

по дисциплине «Криптография и защита информации»

Тема: Изучение шифра AES

Студент гр. 8383 _____ Ларин А.

Преподаватель _____ Племянников А.К.

Санкт-Петербург

2021

Цель работы

Исследовать характеристики шифра AES и финалистов конкурса AES, а также изучить атаку предсказанием дополнения и получить практические навыки работы с шифрами и проведения атаки, в том числе с использованием приложения Cryptool 1 и 2.

1. Исследование преобразований AES

Задание

1. Изучить преобразования шифра AES с помощью демонстрационного приложения из Cryptool 1: *Indiv.Procedures->Visualization…->AES->Rijndael Animation.*
2. Выполнить вручную преобразования для одного раунда и вычисление раундового ключа при следующих исходных данных:
   a. Открытый текст – фамилия_имя (транслитерация латиницей)
   b. Ключ – номер группы_отчество
3. Проверить полученные результаты с помощью приложения-инспектора: *Indiv.Procedures->Visualization…->AES->Rijndael Inspector.*
4. Провести наблюдения в потоковой модели шифра AES с помощью демонстрационного приложения из CrypTool 1 для 0-текста и 0-ключа: *Indiv.Procedures->Visualization…->AES->Rijndael Flow Visualisation*

Выполнение

Описание DES

Шифр AES (Rijndael) работает на основе перестановочно-подстановочной сети (SP-сеть). Обобщенная схема работы алгоритма представлена на рисунке 1.1.

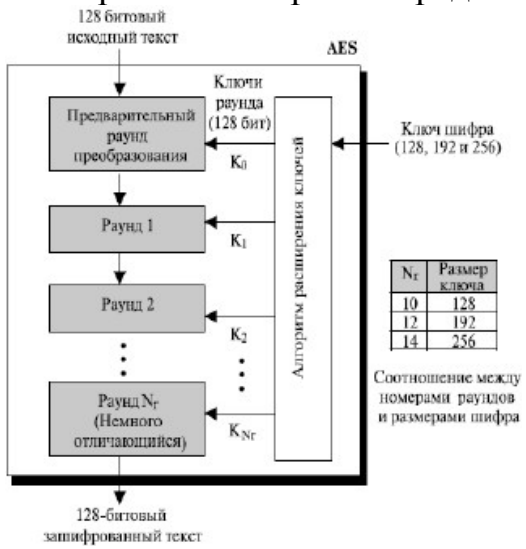


Рис.1.1 – Схема AES

В версии с наименьшей длиной ключа алгоритм AES получает на вход блок открытого текста размером 16 байт и 16 байт ключа. Значения блока записывается в столбцы матрицы состояний размером 4x4 байт.

Процедура расширения ключей ExpandKey создает последовательно (слово за словом) 128 битные раундовые ключи от единственного входного ключа шифра.

После того, как сформированы раундовые ключи, начинается раундовая обработка матрицы состояний. В каждом раунде алгоритма выполняются следующие преобразования, представленные на рисунке 1.2:

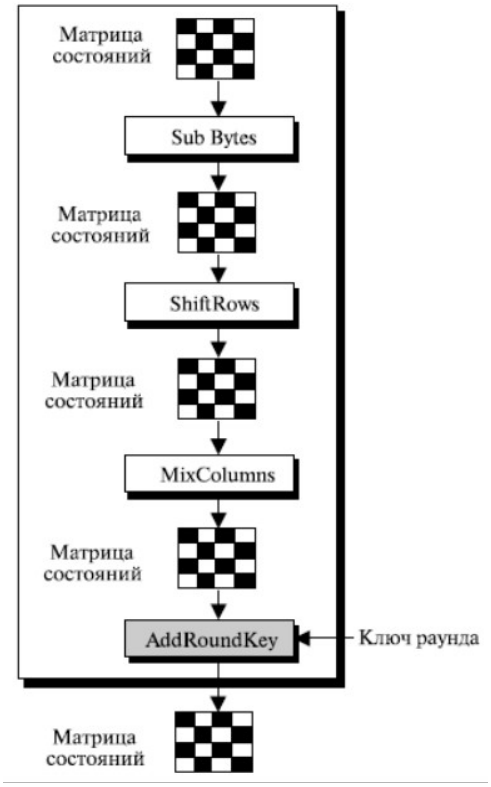


Рис.1.2 – Раундовые преобразования

1. Столбцы матрицы состояний складываются с ключом шифра операцией xor.
2. Полученная матрица состояний проходит через преобразование подстановки SubBytes.
3. Циклический сдвиг влево всех строк матрицы состояний выполняется преобразованием ShiftRows .
4. Смешивание столбцов матрицы состояний путем ее умножения на матрицу констант в конечном поле $GF(2^8)$ выполняет преобразовние MixColumn, а сложение полученных столбцов матрицы состояний с раундовым ключом операцией xor – преобразование AddRoundKey
5. Действия 2-4 повторяются в каждом раунде за исключение последнего.
6. Последний раунд не включает в себя смешивание столбцов.

Расшифровывание выполняется применением обратных операций и раундовых ключей в обратной последовательности.

Для генерации раундовых ключей будем использовать структуру, изображенную на рис. 1.3:

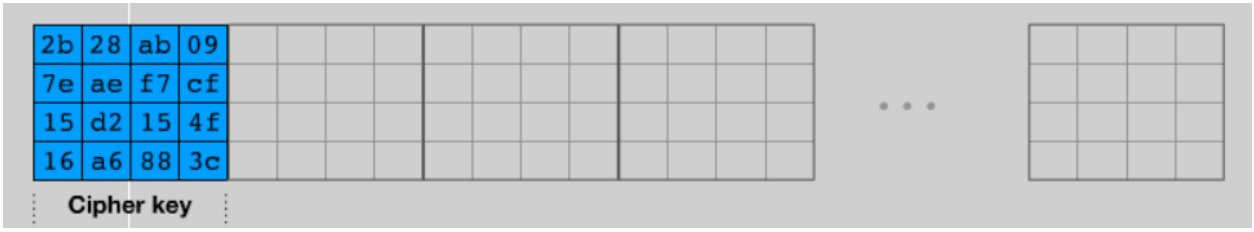


Рис.1.3 – Заполнение раундовых ключей

1. В первый блок заносим ключ шифрования.
2. Для первого слова W_i в следующем блоке выполняем следующие операции:

$W_i = W_{i-1}$

$W_i = RotWord(W_i)$

$W_i = SubBytes(W_i)$

$W_i = W_i \oplus W_{i-4} \oplus Rcon_{\frac{i}{4}}$

где i – номер слова во всей структуре (рис. 1.3), *RotWord* – функция, выполняющая сдвиг слова на один байт влево, *SubBytes* – функция, аналогичная описанной в п.2 раундовых преобразований, *Rcon* – массив из 10 константных слов.

3. Для второго, третьего и четвертого слова в том же блоке поочередно выполняем следующие операции:

$W_i = W_{i-1} \oplus W_{i-4}$

4. Пункты 2-3 выполняем поочередно для всех оставшихся блоков.

Ручной расчет

Открытый текст
larin_anton
Ключ
8383_dmitrievich

Байтовое представление текста T0

```
6c 61 72 69
6e 5f 61 6e
74 6f 6e 00
00 00 00 00
```

Байтовое представление ключа K0

```
38 33 38 33
5f 64 6d 69
74 72 69 65
76 69 63 68
```

T0 xor P0:

```
54 52 4a 5a
31 3b 0c 07
00 1d 07 65
76 69 63 68
```

SubBytes =>
```
20 00 d6 be
c7 e2 fe c5
63 a4 c5 4d
38 f9 fb 45
```

ShiftRows=>
```
20 00 d6 be
e2 fe c5 c7
c5 4d 63 a4
45 38 f9 fb
```

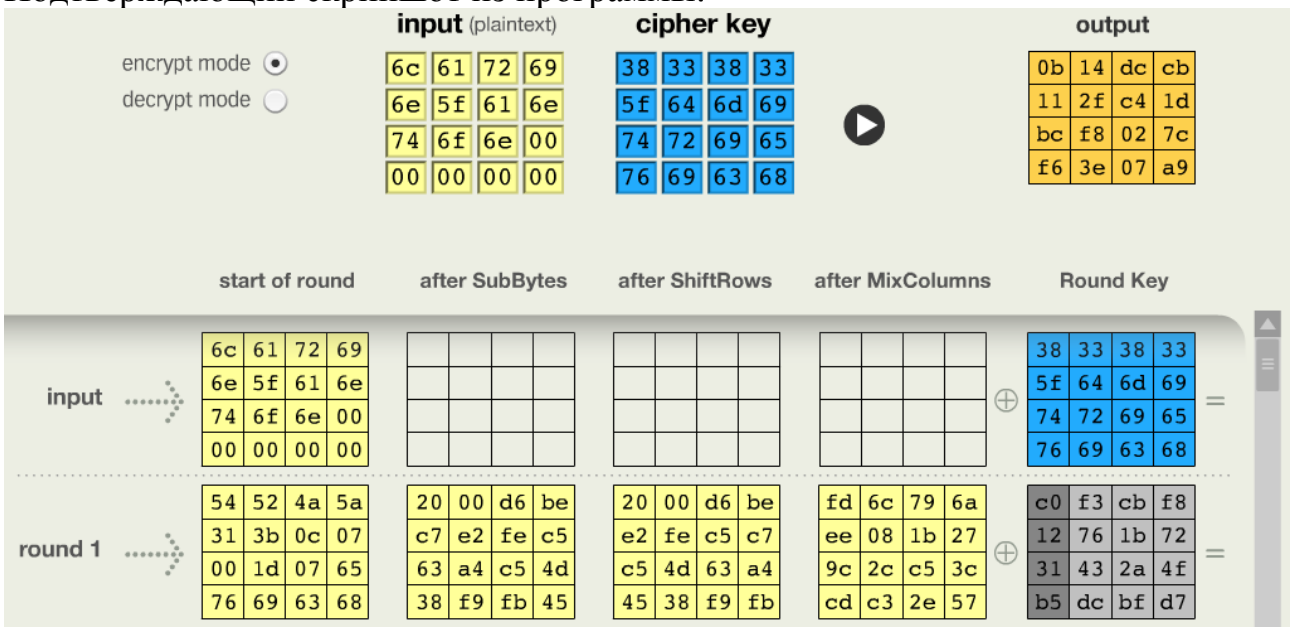
MixColumns=>
```
fd 6c 79 6a
ee 08 1b 27
9c 2c c5 3c
cd c3 2e 57
```

Считаем K1
K0
```
38 33 38 33
5f 64 6d 69
74 72 69 65
76 69 63 68
4-й стоблец
33
69
65
68
Rot word
69
65
68
33
```
SubBytes
f9
4d
45
c3
Xor...
K1
c0 f3 cb f8
12 76 1b 72
31 43 2a 4f
b5 dc bf d7

xor текста с ключем:
54 52 4a 5a

31 3b 0c 07
00 1d 07 65
73 69 63 68

Подтверждающий скриншот из программы:



2. Исследование шифров

Задание

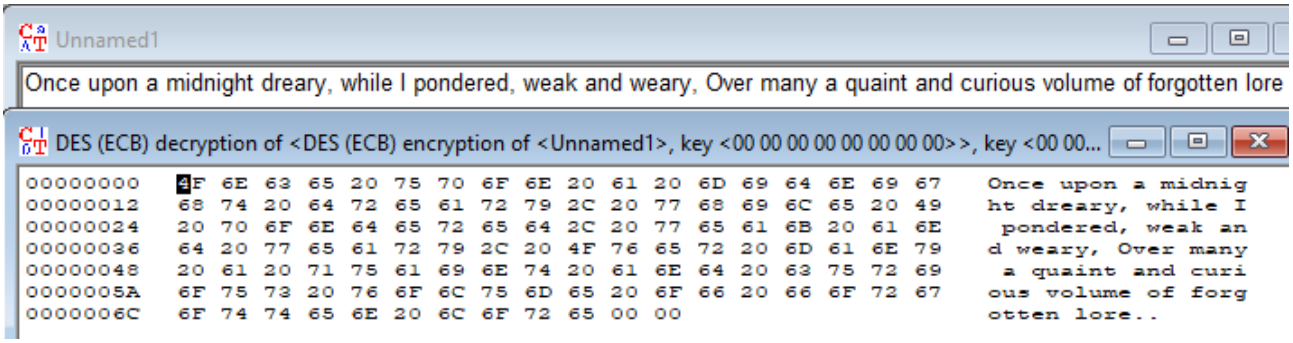
1. Выбрать текст на английском языке (не более 120 знаков).
2. Создать бинарный файл с этим текстом, зашифровав и расшифровав его шифром AES на 0-м ключе.
3. С помощью Cryptool 1 зашифровать с ключом отличным от 0 текст с использованием шифров AES, MARS, RC6, Serpent и Twofish.
4. Приложением из Cryptool 1 вычислить энтропию исходного текста и шифротекстов, полученных в итоге. Зафиксировать результаты измерений в таблице.
5. Приложением из Cryptool 1 оцените время проведения атаки «грубой силы» всех шифров для одного и того же шифротекста в случаях, когда известно n-2, n-4, n-6,..., 2 байт секретного ключа. Зафиксировать результаты измерений в таблице.

Выполнение

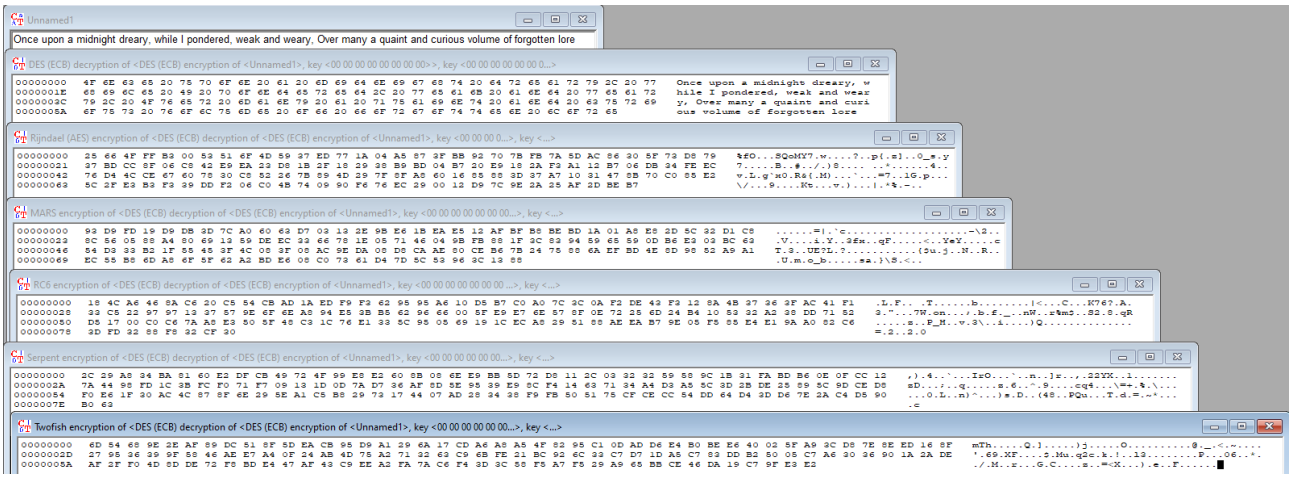
Взят открытый текст длиной 118 символов

```
'Once upon a midnight dreary, while I pondered, weak and weary,
Over many a quaint and curious volume of forgotten lore'
```

В Cryptool 1 получен бинарный вид



Энтропия составила 4.22 при 26-и уникальных символах
Он зашифрован при помощишифров AES, MARS, RC6, Serpent и Twofish



Энтропия приведена в таблице:

Метод	Энтропия
AES	6.49
MARS	6.49
RC6	6.50
Serpent	6.64
Twofish	6.61

Оценка вермени атаки грубой силой (в часах)

Известные байты/Метод	14	12	10	8	6	4	2
AES	-	1.5	1e5	6.7e9	4.5e14	3e19	1.9e24
MARS	-	2.5	1.5e5	1e10	6.7e14	4.38e19	2.9e24
RC6	-	1.5	0.9e5	6.4e9	4.2e14	2.8e19	1.8e24
Serpent	-	4.3	2.8e5	1.8e10	1.2e15	8.1e19	5.3e+24
Twofish	-	2.6	1.7e5	1.2e10	9.6e14	5e+19	3.2e+24

Из таблицы видно, что криптостойкости алгоритмов близки с точностью до порядка, а так же близка энтропия соответствующих шифротекстов. При этом криптостойкость AES примерно равта оной у RC6 и в три раза ниже, чем Serpent

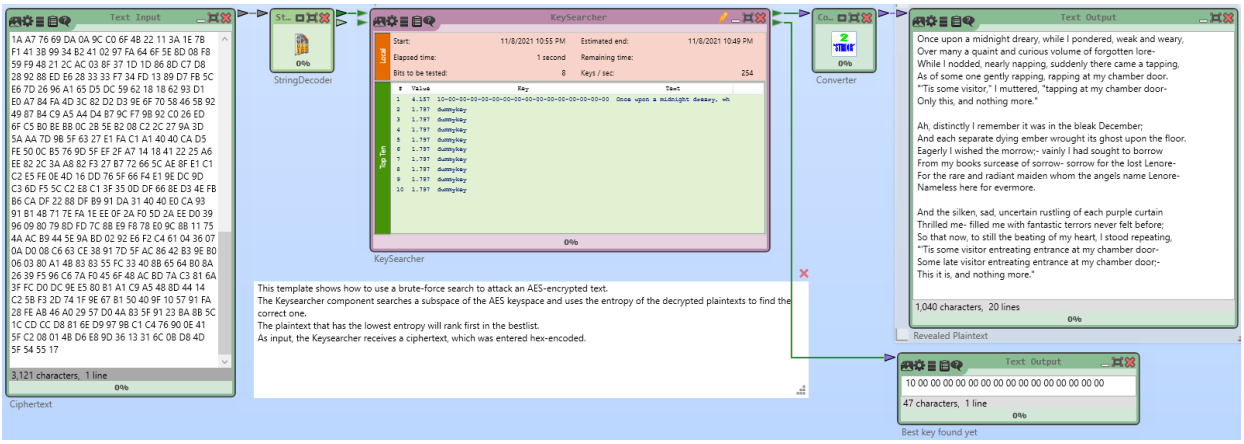
3. Атака «грубой силы» на AES

Задание

1. Найти и запустить шаблон атаки в CrypTool 2: *AES Analysis using Entropy*.
2. Выбрать открытый текст (примерно 1000 знаков) и загрузить его в шаблон.
3. Провести атаку «грубой силы» когда известно n-2, n-4, n-6 байт секретного ключа, используя в качестве оценочной функции энтропию и задействовав 1 ядро процессора. Зафиксировать затраты времени.
4. Выполнить атаку повторно с средним и максимальным количеством процессорных ядер. Зафиксировать затраты времени.
5. Сформировать текст с произвольным сообщением в формате «DEAR SIRS message THANKS» и загрузить его в шаблон.
6. Провести атаку «грубой силы» когда известно n-2, n-4, n-6 байт секретного ключа, используя в качестве оценочной функции словосочетание DEAR SIRS задействовав 1 ядро процессора. Зафиксировать затраты времени.
7. Выполнить атаку повторно с средним и максимальным количеством процессорных ядер. Зафиксировать затраты времени.

Выполнение

Был взят открытый текст из примерно 1000 символов и загружен в схему атаки в Cryptool 2



Была проведена атака грубой силой с разной известной частью ключа и разным количеством ядер

Известные байты/Ядра	14	12	10
1	-	1.2	8.5e4
2	-	0.4	3e4
4	-	0.25	1.6e4

Видно пропорциональное ускорение пир использовании нескольких ядер, однако результаты остаются в пределах одного порядка т. е. при большом количестве вариантов время не снижается существенно.

Взят открытый текст «DEAR SIRS VISITORS THANKS»
В качестве оценочной функции взято вхождение подстроки «DEAR SIRS»
Была проведена атака грубой силой с разной известной частью ключа и разным количеством ядер

Известные байты/Ядра	14	12	10
1	-	0.4	2.8e4
2	-	0.15	1e4
4	-	0.08	6e3

Атака с использованием вхождения строки более эффективна, но всё таки не даёт значимого преимущества при большом количестве вариантов

4. Атака предсказанием дополнения на шифр AES в режиме CBC (Padding Oracle Attack)

Задание

1. Найти и запустить шаблон атаки в CrypTool 2: Padding Oracle Attack on AES.
2. Подготовьтесь к атаке теоретически:
   a. Изучите комментарии к шаблону
   b. Изучите публикацию
3. Внедрите во второй блок исходного текста коды символов своего имени.
4. Выполните 3 фазы атаки и сохраните итоговые скриншоты по окончанию каждой фазы.

5. Убедитесь, что атака удалась.

Выполнение

Атака предсказанием дополнения на шифр AES в режиме CBC (Padding Oracle Attack)

При проведении этой атаки предполагается, что нарушитель может модифицировать и отправлять зашифрованное сообщение серверу для расшифровки, а также распозновать ответы сервера корректности дополнения последнего блока. Десшифровка сообщения нарушителем начинается с последнего блока шифротекста.
Рассмотрим расшифровку блока C_{i+1} .
1. Формируем R – все биты, кроме последнего, случайные значения. Перебираем байт R_n от 0x00 до 0xFF, каждый раз посылая на сервер $[R \| C_{i+1}]$. Если при некотором R n сервер «одобряет», то $T_n = 01, S_n = R_n \oplus 0x01, p_n = S_n \oplus C_n$. Схема первого этапа представлена на рисунке 5.3.

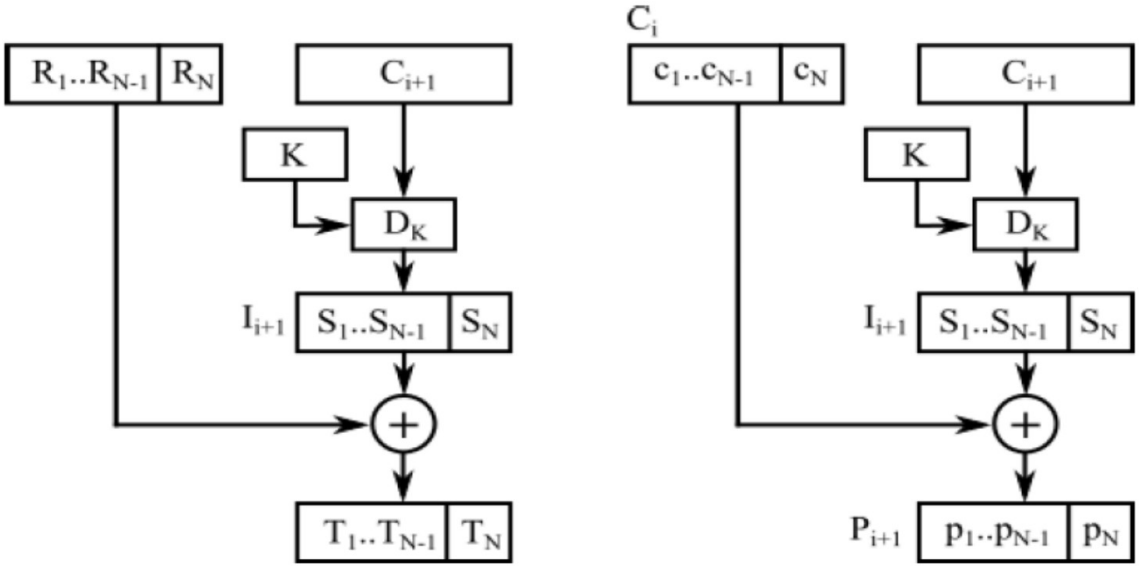


Рисунок 5.3

P_i – открытый текст, C_i – шифротекст, I_i – промежуточное состояние , K – ключ, D_K – функция расшифровки, T_i – формируемое дополнение.

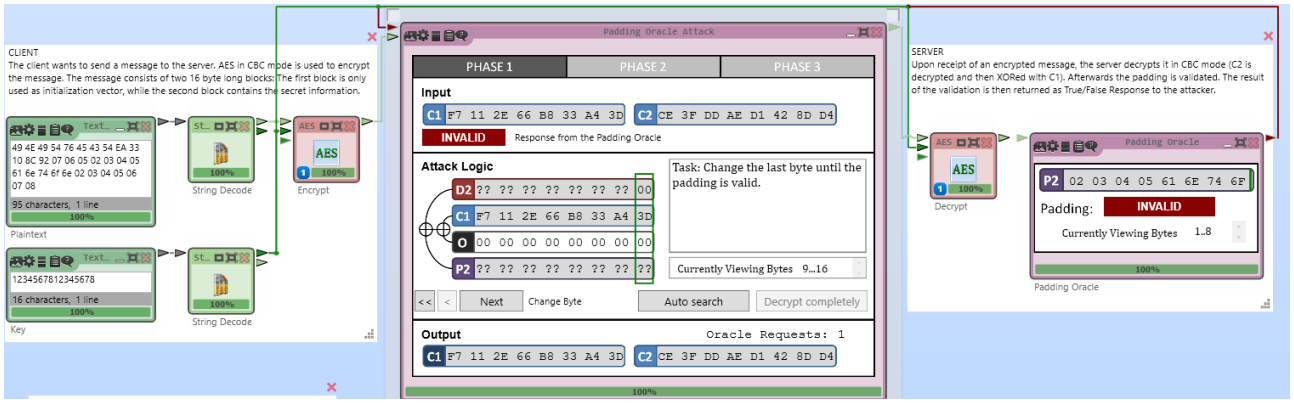
2. Формируем R – все биты, кроме двух последних, случайные значения. $R_n = S_n \oplus 0x02$, чтобы $T_n = 02$. Перебираем байт R_{n-1} от 0x00 до 0xFF, каждый раз посылая на сервер $[R \| C_{i+1}]$. Если при некотором R_{n-2} сервер «одобряет», то $T_{n-1} = 02, S_n = R_{n-1} \oplus 0x02, p_{n-1} = S_{n-1} \oplus C_{n-1}$.
На третьем шаге пытаемся получить дополнение 030303, на четвертом – 04040404. После N шагов получаем полностью блок p_{i+1} .

В CryoTool 2 атака предсказанием дополнения реализована в три фазы:
1. Фаза 1. Нахождение длины дополнения, т.е. последний байт

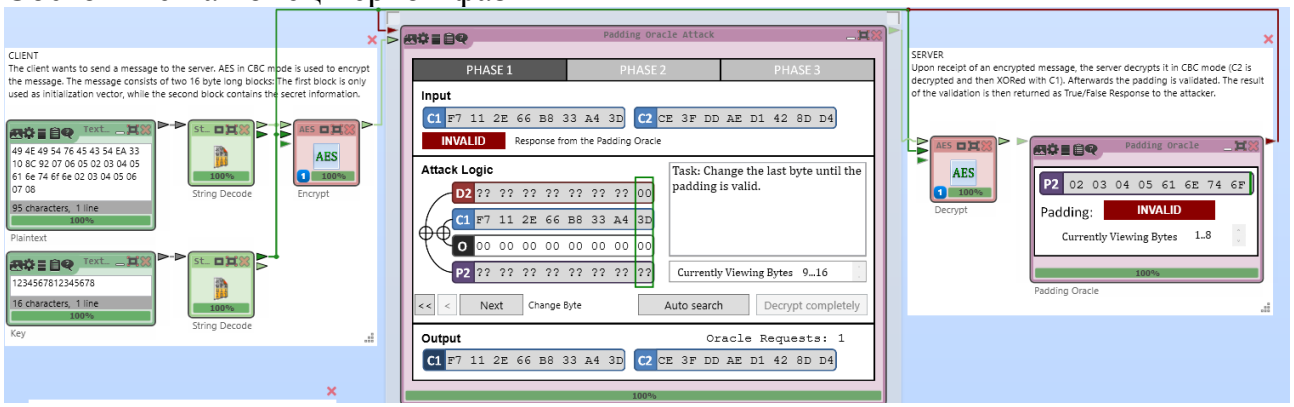
2. Фаза 2. Подбор дополнения
3. Фаза 3. Расшифровка текста

Во второй блок были внедрены символы anton
`61 6e 74 6f 6e`
Ключ оставлен по умолчанию

Исходная конфигурация:

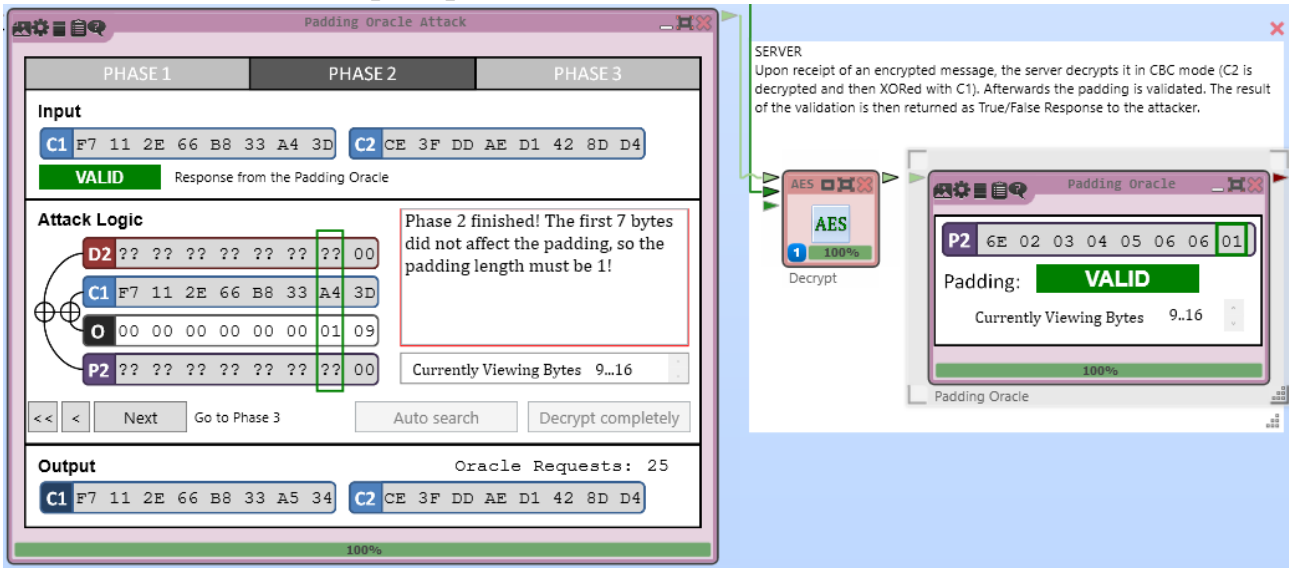


В первой фазе был найден последний байт, при котором получается верное дополнение (в формируемом дополнении Т получили 01 или 0202 и т.д.на конце)
Состояние на конец первой фазы

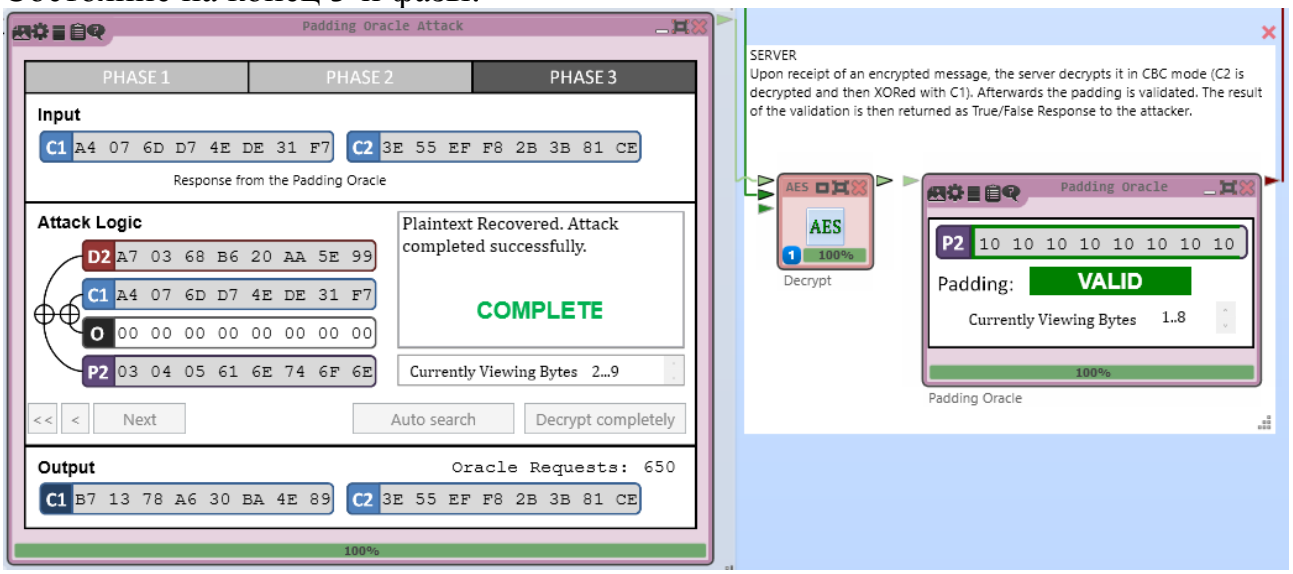


Во второй фазе ищем первый байт дополнения меняя байты слева напаво, пока сервер не ответит, что дополнение не валидное. Это означает, что мы нашли первый байт дополнения, следовательно его длину. В данном случае дополнение окащалось длины 1.

Состояние на конец второй фазы:



Это позволяет расшифровать последний байт
Находим последний байт блоке D2 = 34 xor 01 = 35
P = D2 ^ C1 = 35 xor 3D = 08, что верно
Далее в фазе 3 идет побайтовая расшифровка подбором R так, чтобы получить «добро» от сервера по поводу доплнения
Состояние на конец 3-й фазы:



В расшифрованом тексте видны искомые байты anton
`61 6e 74 6f 6e`
В целом текст аналогичен исходному. Атака удалась

Выводы.

1. Был исследован шифр AES. Был изучен алгоритм его работы, воспроизведён вручную, результат сравнен с машинной обработкой. Результаты совпали.
2. Была изучены другие финалисты конкурса AES: Rijndael, MARS, RC6, Serpent, Twofish.

   Был выбран текст, после чего зашифрован всеми шифрами. По итогу произведено сравнение параметров энтропии и сложность атаки грубой силой при различной известной части улюча. Значение энтропии получились очень похожими.

   Результаты атаки были немого различны, так Serpent оказался примерно в три раза более криптостойким, однако разница во всех случаях оказалась менее порядка, т. е. AES(Rijndael) существенно не уступает конкурентам
3. Была произведена атака грубой силой на AES при оценочной функции энтропии и известного словосочетания, а так же при отхном количестве ядер и известных байт ключа

   В результата количество ядер дало пропорциональный прирост к скорости. оценочная функция по словосочетанию показала немного лучший результат. Однако никакое из этих преимуществ не является решающим, т. к. при малой известной длине ключа они не взламываемы за разумное время вне зависимости от количества ядер.
4. Была изучена Атака предсказанием дополнения на шифр AES в режиме CBC (Padding Oracle Attack), основанная на знании правил формирования паддинга, а так же аозможности отправлять серверу сообщения для расшифровки

   Изучен теоретический принцип её работы, а так же реализация в Cryptool 2, с разделением на три фазы.

   В первой фазе найден байт первого блока, при котором получается верное дополнение

Во второй фазе последовательным изменением байт слева направо найдена длина доплнения. После чего в третьей фазе, инкрементируя дополнение и поиск валидного по ответам сервера, побайтово найден весь промежуточный блок, который при XOR с первым блоком шифротекста (принцип CBC) дал открытый текст. Открытый текст совпадал с исходным, в т.ч. содержал буквы имени.