Белорусский государственный университет информатики и радиоэлектроники

Кафедра информатики

# Лабораторная работа № 1

# Симметричная криптография. Двойной и тройной DES.

Выполнила студентка гр. 653502: Сулима М.Ф.

Проверил ассистент КИ: Артемьев В. С.

Минск, 2019

**Введение**

Стандарт шифрования данных DES (DATA ENCRYPTION STANDARD) – блочный шифр с симметричными ключами, разработан Национальным Институтом Стандартов и Технологии (NIST – National Institute of Standards and Technology).

Для шифрования DES принимает 64-битовый открытый текст и порождает 64-битовый зашифрованный текст и наоборот, получив 64 бита зашифрованного текста, он выдает 64 бита расшифрованного. В обоих случаях для шифрования и дешифрования применяется один и тот же 56-битовый ключ.

Чтобы увеличивать криптостойкость DES, появляются несколько вариантов: double DES (2DES), triple DES (3DES).

Методы 2DES и 3DES основаны на DES, но увеличивают длину ключей (2DES — 112 бит, 3DES — 168 бит) и поэтому увеличивается криптостойкость.

Схема 3DES имеет вид $DES(k_3, DES(k_2, DES(k_1, M)))$, где $k_1, k_2, k_3$ ключи для каждого шифра DES. Это вариант известен как в EEE, так как три DES операции являются шифрованием. Существует 3 типа алгоритма 3DES:

- DES-EEE3: Шифруется три раза с 3 разными ключами.
- DES-EDE3: 3DES операции шифровка-расшифровка-шифровка с 3 разными ключами.
- DES-EEE2 и DES-EDE2: Как и предыдущие, за исключением того, что первая и третья операции используют одинаковый ключ.

В рамках лабораторной работы необходимо реализовать программные средства шифрования и дешифрования при помощи алгоритмов двойной и тройной DES.

**Блок-схема алгоритма**

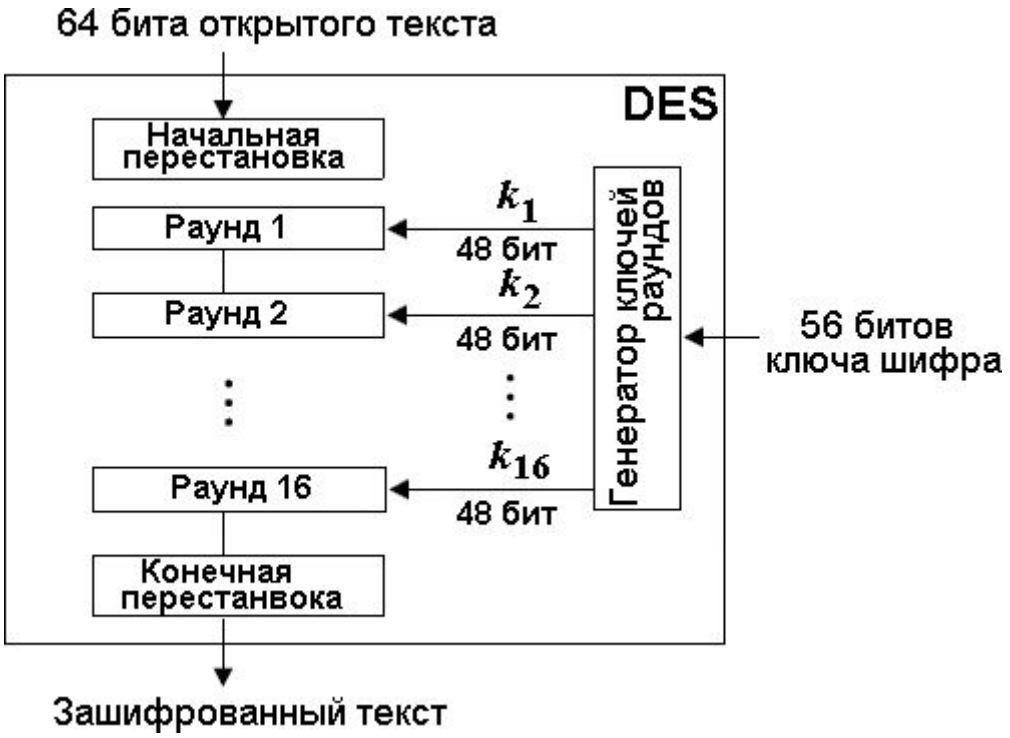


Рис.1. Блок-схема DES

Раунды DES

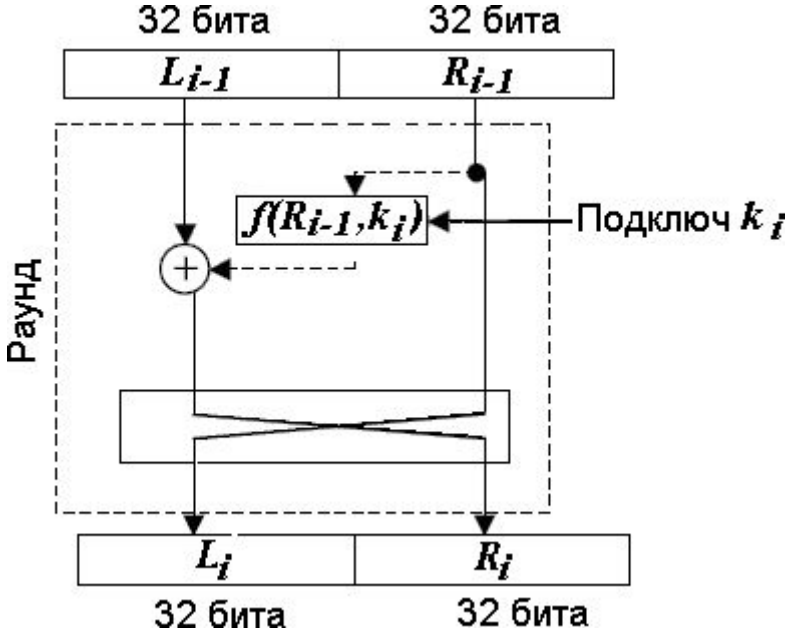

Рис.2. Раунды DES

Функция DES

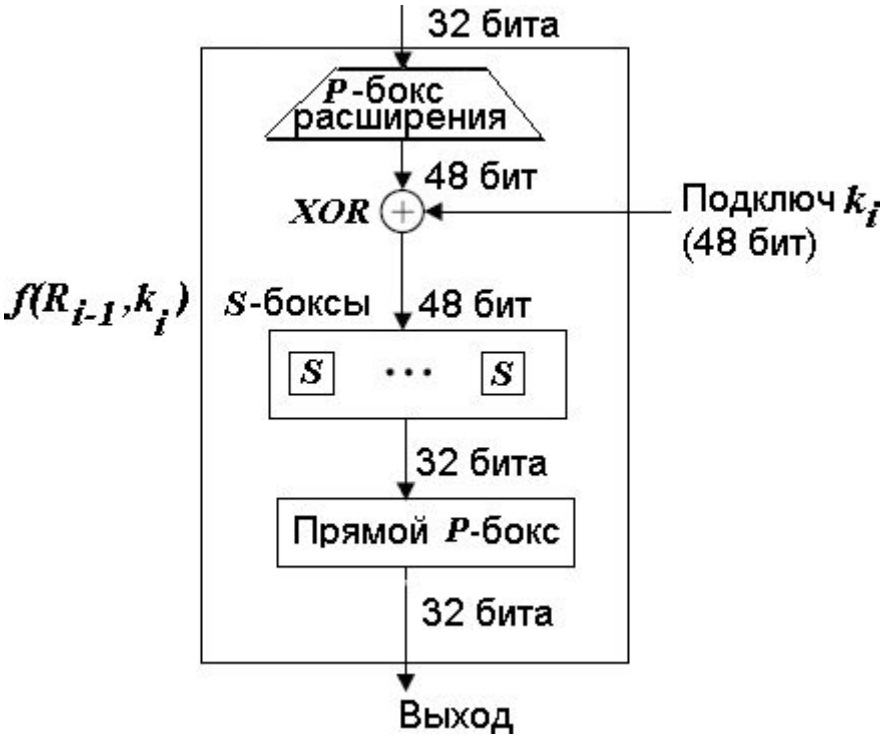


Рис.3. Функция DES

**Пример работы программы**

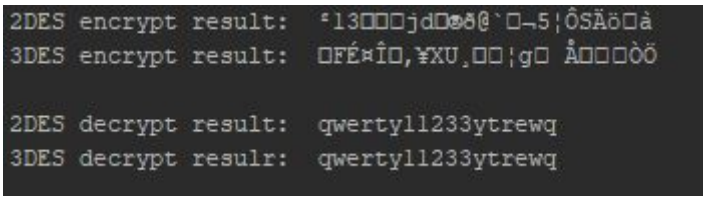


Рис.4. Пример работы

**Код программы**

```
def perform_rounds(m_array, is_encrypt):
   left_part = m_array[:32]
   right_part = m_array[32:]
   if is_encrypt:
       for i in range(0, 16):
           temp_array = right_part
                 right_part = [k ^ l for k, l in zip(left_part,
 perform_round(right_part, i))]
           left_part = temp_array
       return right_part + left_part
   else:
       for i in range(16, 0, -1):
           temp_array = right_part
                 right_part = [k ^ l for k, l in zip(left_part,
 perform_round(right_part, i - 1))]
           left_part = temp_array
   return right_part + left_part
def ip_text(text, is_encrypt):
   perm_arr = IP if is_encrypt is False else InvP
   arr = []
   for i in range(0, len(perm_arr)):
       arr.append(text[perm_arr[i] - 1])
   return arr
def encrypt(key_text, plain_text):
   s = ""
   create_keys(key_text)
       text_array = [plain_text[i:i + 8] for i in range(0,
 len(plain_text), 8)]
   if len(plain_text) % 8 != 0:
                          text_array[len(text_array) - 1] =
 str(text_array[len(text_array) - 1]).ljust(8, " ")
   for i in range(0, len(text_array)):
                                              inv_perm_array =
 ip_text(perform_rounds(ip_text(str_to_bit_array(text_array[i]),
 False), True), True)
       s = s + bit_array_to_str(inv_perm_array)
   return s
def decrypt(encrypted_text, key_text):
   create_keys(key_text)
   s = ""
      text_array = [encrypted_text[i:i + 8] for i in range(0,
 len(encrypted_text), 8)]
   for i in range(0, len(text_array)):
                                                decrypt_part =
 ip_text(perform_rounds(ip_text(str_to_bit_array(text_array[i]),
 False), False), True)
       s = s + bit_array_to_str(decrypt_part)
   return s
```

## Вывод

Сам по себе алгоритм DES уже не является криптостойким, т.к. силами современной вычислительной техники его вполне можно взломать. С попыткой увеличения криптостойкости, используя двойной DES, была выявлена слабость, называемая “встреча посередине”, что тоже делает алгоритм уязвимым. Что касается тройного DES, то на данный момент его можно считать криптостойким. Для успешной атаки на 3DES потребуется около $2^{32}$ бит известного открытого текста, $2^{113}$ шагов, $2^{90}$ циклов DES-шифрования и $2^{88}$ бит памяти. На данный момент это непрактично, и, по оценкам НИСТ, алгоритм с выбором трех различных ключей должен остаться надежным до 2030-х.