Cel zadania

Celem zdania jest implementacja algorytmu szyfrowania AES (Advanced Encyrption Standard).

Działanie algorytmu

Algorytm składa się z 2 głównych części:

- Key Expansion
- Szyfrowania/Deszyforwanie.

Key Expansion

W zależnośći od długości klucza głównego (128, 192 lub 256 bitów) generowane są klucze do kolejnych rund algorytmu.

Ilość wygenerowanych kluczy:

- klucz 128 bitów → 10 wygenerowanych + klucz szyfrowania
- klucz 192 bitów → 12 wygenerowanych + klucz szyfrowania
- klucz 256 bitów → 14 wygenerowanych + klucz szyfrowania

Dla klucza głównego 128 bitów algorytm rozszerzania klucza wyglada następująco:

```
1. Weź pusta tablice klucz[] o długości 176 (16 * 11)
2. Ustaw pierwsze 16 bajtów klucz[] jak bajty klucza szyfrowania
3. Ustaw i := 1 (numer rundy)
4. Ustaw c := 16 (liczba bajtów klucza)
5. Ustaw temp[] := bajty z klucz[] od c-4 do c
6. Jeżeli c MOD 16 = 0 to krok 7 inaczej krok 10
7. Obróć temp w lewo o jeden bajt
    [a0, a1, a2, a3] => [a1, a2, a3, a0]
8. Wykonaj operacje podmienienia temp na wartości z AES SBox
9. Wykonaj operacje XOR na pierwszym bajcie temp wykorzystując RCON[i]
10. Wykonaj operacje XOR na koljenych 4 bajtach temp[] oraz klucz[] od c-16 i
zapisz wynik do klucz[] na pozycjach od c do c+3
11. Ustaw c:=c+4
12. Jeżeli c < 176 to krok 5 inaczej zakończ
```

W przypadku klucza głównego o długości 128 lub 192 bitów, algortym jest podobny. Dla klucza 192 bity alogrytm różni się długością tablicy klucz, wynosi ona 208 (16 * 13) oraz zamiast 16 używamy 24 (długość klucza w bajtach).

Dla klucza szyfrowania o długości 256 bitów oprócz różnic zaznaczonych w przypadku klucza 192 bity(długość tablic klucz wynosci 240 bajtów, wykorzystujemy wartość 32) co 16 bajtów na tablicy temp wykonujmy operacje AES SBox.

Szyfrowanie/Deszyfrowania

Aes jest algortymem blokowym o długości bloku 128 bitów.

Algorytm polega na wykonywania określonej liczby rund na każdym z bloków. Liczba rund jest uwarunkowana długością klucza.

Pierwszym krokiem jest podzielenie danych na 16 bajtowe bloki oraz dodanie paddingu.

Poszczególne kroki algorytmu wykonywane są na danych zapisanych w postacji State Array.

$$\begin{bmatrix} a_0 & a_1 & a_2 & \dots & a_{15} \end{bmatrix}$$

$$\downarrow$$

$$\begin{bmatrix} a_0 & a_4 & a_8 & a_{12} \\ a_1 & a_5 & a_9 & a_{13} \\ a_2 & a_6 & a_{10} & a_{14} \\ a_3 & a_7 & a_{11} & a_{15} \end{bmatrix}$$

Algorytm szyfrowania bloku:

```
1. Zamień wchodzącą tablicę danych na state array zgodnie z powyższym schematem
2. Wykonaj operacje Add Round Key przy użyciu zerowego klucza
3. Wykonaj operacje Sub-bytes
4. Wykonaj operacje Shift Rows
5. Wykonaj operacje Mix Columns
6. Wykonaj operacje Add Round Key
7. Powtórz kroki od 3 do 6 odpowiednią liczbę razy (liczba rund).
!!! Ostatnia runda nie zawiera kroku Mix Columns !!!
```

Add Round Key:

Zamieniamy klucz na state array i wykonujemy XOR na kolejnych bajtach bloku i klucza.

Sub-bytes:

Zamień poszczególne bajty bloku zgodnie z AES SBox

Shift Rows:

Przesuń w lewo o n (licząc od zera) n-ty wiersz (tak samo jak w 7 kroku key expansion).

Mix Columns:

Przemnoż state array przez macierz:

$$\begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix}$$

Dodawanie wykonujemy modulo 2.

Deszyfrowanie odbywa się w odwrotnej kolejności.

Algorytm deszyfrowania bloku:

```
1. Zamień wchodzącą tablicę danych na state array zgodnie z powyższym schematem
2. Wykonaj operacje Add Round Key przy użyciu ostatnie klucza
3. Wykonaj operacje Inv Shift Rows
4. Wykonaj operacje Inv Sub-bytes
5. Wykonaj operacje Add Round Key
6. Wykonaj operacje Inv Mix Columns
7. Powtórz kroki od 3 do 6 odpowiednią liczbę razy (liczba rund).
!!! Ostatnia runda nie zawiera kroku Mix Columns !!!
```

Inv Shift Rows

Odwracamy operacje shift rows poprzez przesuwanie w prawo analogicznie jak w przypadku shift rows.

Inv Sub-bytes

Zamiast AES SBox używamy Inverse AES SBox

Inv Mix Columns

Przemnoż state array przez macierz:

$$\begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix}$$

Dodawanie wykonujemy modulo 2.