

# *Wojskowa Akademia Techniczna im. Jarosława Dąbrowskiego*

Laboratorium z przedmiotu:  
Wprowadzenie do Kryptologii

Sprawozdanie z ćwiczenia laboratoryjnego nr 3:  
**Funkcje skrótu**

Prowadzący:  
mgr inż. Marta Turowska

**Wykonał:** Radosław Relidzyński

**Grupa:** WCY20IY4S1

**Data laboratoriów:** 11.05.2021 r.

## Spis treści

|    |   |   |
|----|---|---|
| A. | Treść zadania .....                         | 2 |
| B. | Użyte komendy .....                         | 2 |
| C. | Szyfrowanie pliku.....                      | 2 |
|    | Użyte polecenia: .....                      | 2 |
|    | Zawartość pliku „first_ecb.txt” .....       | 3 |
|    | Zawartość pliku „first_cbc.txt” .....       | 3 |
| D. | Szyfrowanie pliku ze zmienionym bitem ..... | 4 |
|    | Użyte polecenia: .....                      | 4 |
|    | Zawartość pliku „second _ecb.txt” .....     | 4 |
|    | Zawartość pliku „second _cbc.txt” .....     | 5 |
| E. | Podsumowanie .....                          | 5 |

## A. Treść zadania

Przy pomocy openssl wykorzystując szyfr AES zaszyfrować w trybie ECB plik, a następnie zaszyfrować w trybie CBC. Następnie zmienić 1 bajt w pliku i ponownie zaszyfrować szyfrem AES w trybie ECB oraz CBC. Porównać otrzymane wyniki i wyciągnąć wnioski. Podać możliwe zastosowania dla tych szyfrowań i ocenić bezpieczeństwo obydwu. Jako klucz użyć swój numer indeksu.

## B. Użyte komendy

W programie openssl.exe wykonuję poniższe komendy:

aes-256-ecp, aes-256-cbc

Jako argument będę podawał plik wejściowy

-in {file\_with\_path}

Jako kolejne argumenty będę podawał ścieżkę do pliku wyjściowego

-out {file\_with\_path}

Jako klucz szyfrowania podam swój numer indeksu, czyli

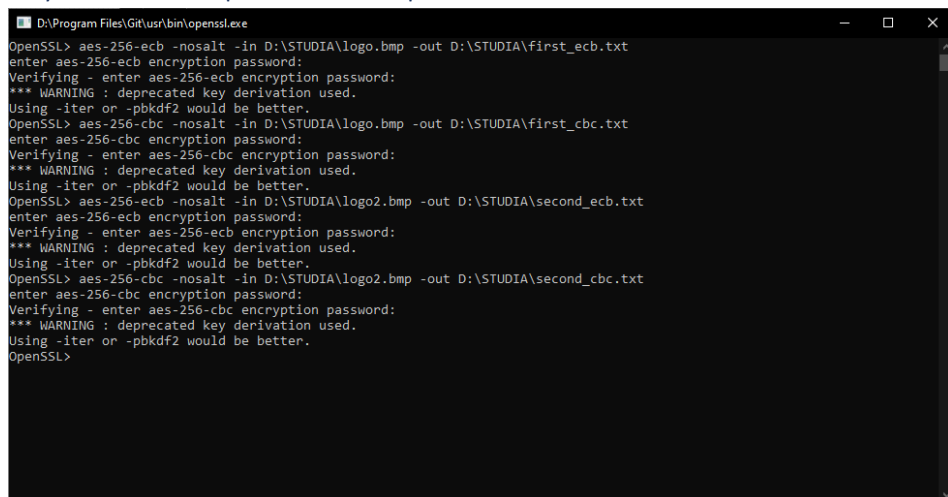
76836

Aby móc dobrze porównać rezultat, dodam specjalną flagę

-nosalt

Następnie będę analizować pliki tekstowe zawierające zaszyfrowane wiadomości

## Wywołanie odpowiednich poleceń



```

D:\Program Files\Git\usr\bin\openssl.exe
OpenSSL> aes-256-ecb -nosalt -in D:\STUDIA\logo.bmp -out D:\STUDIA\first_ecb.txt
enter aes-256-ecb encryption password:
Verifying - enter aes-256-ecb encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
OpenSSL> aes-256-cbc -nosalt -in D:\STUDIA\logo.bmp -out D:\STUDIA\first_cbc.txt
enter aes-256-cbc encryption password:
Verifying - enter aes-256-cbc encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
OpenSSL> aes-256-ecb -nosalt -in D:\STUDIA\logo2.bmp -out D:\STUDIA\second_ecb.txt
enter aes-256-ecb encryption password:
Verifying - enter aes-256-ecb encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
OpenSSL> aes-256-cbc -nosalt -in D:\STUDIA\logo2.bmp -out D:\STUDIA\second_cbc.txt
enter aes-256-cbc encryption password:
Verifying - enter aes-256-cbc encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
OpenSSL>
```

## C. Szyfrowanie pliku

Użyte polecenia:

aes-256-ecb -nosalt -in D:\STUDIA\logo.bmp -out D:\STUDIA\first\_ecb.txt

aes-256-cbc -nosalt -in D:\STUDIA\logo.bmp -out D:\STUDIA\first\_cbc.txt

## Zawartość pliku „first\_ecb.txt”

### Zawartość pliku „first cbc.txt”



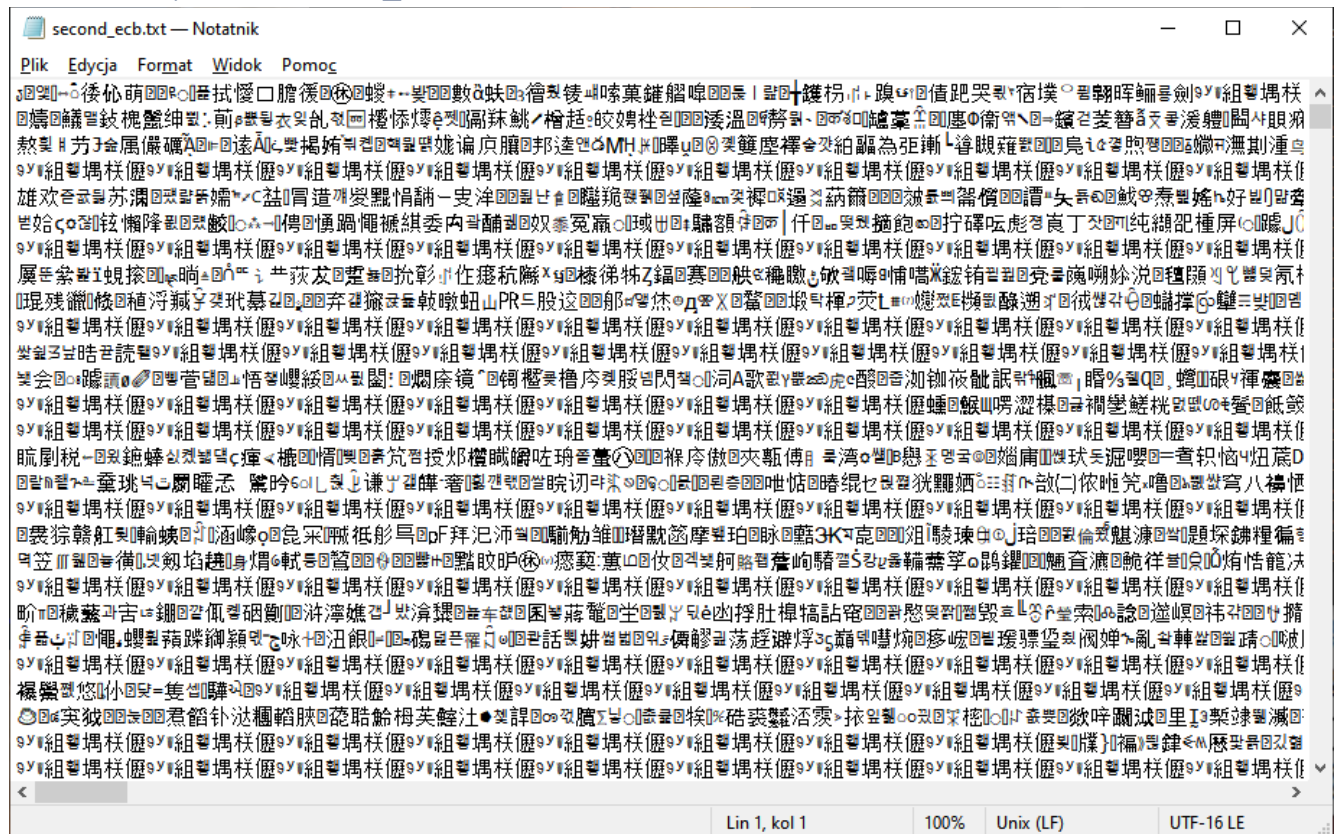
#### D. Szyfrowanie pliku ze zmienionym bitem

Użyte polecenia:

```
aes-256-ecb -nosalt -in D:\STUDIA\logo2.bmp -out D:\STUDIA\second_ecb.txt
```

```
aes-256-cbc -nosalt -in D:\STUDIA\logo2.bmp -out D:\STUDIA\second_cbc.txt
```

### Zawartość pliku „second\_ecb.txt”



## Zawartość pliku „second\_cbc.txt”



## E. Podsumowanie

### Tryb ECB

Szyfrowanie w trybie ecb polega na szyfrowaniu wiadomości w sposób blokowy. Każdy blok jest szyfrowany i odszyfrowywany oddzielnie. Dobrze sprawdza się do szyfrowania równoległego, gdzie 2 procesy mogłyby równolegle odszyfrowywać tę samą wiadomość.

### Tryb CBC

Szyfrowanie w trybie cbc natomiast polega na dodaniu wektora inicjującego, przy pomocy bramki XOR szyfruje pierwszy blok tekstu. Każdy następny blok dostaje wektor na podstawie ostatniego wyniku szyfrowania. Przez to każdy kolejny blok szyfrogramu jest zależny od wyniku poprzedniego, przez co nie ma możliwości odszyfrowania jakiegokolwiek części, jeśli nie odszyfruje się całej poprzedzającej ją części szyfrogramu.

### Ocena bezpieczeństwa

Pod względem bezpieczeństwa, szyfrowanie CBC jest znacznie bezpieczniejszym trybem szyfrowania od ECB, ze względu na dodatkową mechanikę, którą posiada. Dzięki temu, że to my podajemy wektor inicjujący, możemy podać losową wartość rozbudowującą szyfrowanie. Dodatkowo, przez opieranie szyfrowania na poprzednich wynikach, nie ma możliwości rozszyfrowania wiadomości w wybranym ze środka fragmencie.