Mikołaj Stępniak 236659

Mateusz Przybył 236630

**Podstawy Kryptografii**

**Zestaw I program 1**

1. **Cel zadania**

Celem zadania było napisanie programu szyfrującego/deszyfrującego dane wprowadzone przez użytkownika lub z pliku wykorzystując algorytm DES.

Do wykonania zadania posłużyliśmy się językiem Java w środowisku IntelliJ IDEA Community Edition 2021.2.2. Użytkownik ma opcję wprowadzenia wartości heksadecymalnej klucza, wpisania własnoręcznie tekstu jawnego poddanego później szyfrowaniu/deszyfrowaniu wedle wyboru, jak i wybrania opcji pobrania go z pliku.

1. **Algorytm DES**
2. Tekst jawny dzielony jest na bloki zawierające po 64 bity, w razie konieczności dopełniane znakiem „~”, następnie dodawane do Listy.
3. Dokonywane jest tworzenie 16 48-bitowych podkluczy za pomocą funkcji generateRoundKeys(key), gdzie key jest kluczem w postaci heksadecymalnej wprowadzonej przez użytkownika:
4. Aby otrzymać 56-bitowy klucz permutujemy 64-bitowy klucz wprowadzony przez użytkownika za pomocą tablicy PC1 (Permuted Choice 1).
5. 56-bitowy klucz dzielimy na dwie części – połowę lewą i połowę prawą (każda po 28 bitów)
6. Prawa i lewa połowa przesuwana jest o 1 bądź 2 bity (w zależności od tablicy iterationShift) w lewo, z wyjątkiem 1 bitu (bądź 2 bitu), który jest umieszczany na końcu bloku.
7. Obie części łączymy w jeden 56-bitowy blok.
8. Dokonujemy permutacji 56-bitowego bloku z tablicą PC2 (Permuted Choice 2)w celu otrzymania bloku 48-bitowego (podklucza).
9. Otrzymany podklucz umieszczany jest w tablicy roundKeys o 16 wierszach.
10. Powtarzamy kroki 2., 3., 4., 5., 6., jeszcze 15 razy w celu uzyskania 16 48-bitowych kluczy wymaganych do działania algorytmu.
11. Bloki poddajemy permutacji z tablicą IP (Initial PermutationTable).
12. Blok dzielimy na dwie części – lewą i prawą część, każda po 32 bity.
13. Wykonujemy 16 rund szyfrowania zgodnie z algorytmem DES (przy pomocy 16 podluczy, każda iteracja ma odpowiedni klucz np. runda 1 – wykorzystujemy 1-szy podklucz, runda 2 – wykorzystujemy 2-gi podklucz):
14. 32-bitową prawą stronę poddajemy permutacji z tablicą E i otrzymujemy prawą stronę o wielkości 48 bitów.
15. Zmienioną stronę prawą poddajemy operacji XOR z podkluczem 48-bitowym (odpowiednim dla rundy).
16. Dokonujemy permutacji z tablicą SBox (S-Box) zmienioną stroną prawą, w ten sposób zmienia ona rozmiar do 32 bitów.

* 48 bitowy blok prawej strony dzielimy na 8 bloków po 6 bitów (dla każdego bloku istnieje odpowiadająca mu tablica w tablicy SBox).
* Bity położone na zewnątrz (bity skrajne np. 011001) po połączeniu i przeliczeniu na postać dziesiętną będą wskazywać wiersz SBox, a bity wewnętrzne (np. 011001) po przeliczeniu na wartość dziesiętną będą wskazywać kolumnę SBox.
* Na podstawie operacji (wiersz · 16) + kolumna określany jest, który element z tablicy SBox ma być przeliczony na postać binarną i wstawiony w miejsce 6 bitowego bloku

1. Następuje permutacja 32-bitowej strony prawej z tablicą P (P-Box).
2. Dokonywana jest operacja XOR strony lewej z zmienioną przez poprzednie kroki stroną prawą.
3. Otrzymany ciąg bitów jest zapisywany w miejscu prawej strony, a niezmieniona postać prawej strony (z wcześniej wykonanej kopii) jest zapisywana w miejscu lewej strony.
4. Obie strony łączone są w jeden 64-bitowy blok i poddawane permutacji z tablca reversedIP
5. Końcowo otrzymane 64-bitowe bloki łączymy ze sobą – tworzą one nasz szyfrogram.

Deszyfrowanie odbywa się w sposób analogiczny z wyjątkiem, że podczas „16 rund” podklucze przekazywane są w odwrotnej kolejności (np. runda 1 – 16-sty podklucz, runda 2 – 15-sty podklucz)

1. **Wnioski końcowe**

* Wadą algorytmu szyfrowania DES jest możliwość wystąpienia tzw. kluczy słabych, czyli kluczy przy których wykorzystaniu powstałe podklucze wykorzystane w kolejnych rundach będą takie same co ułatwi rozszyfrowanie naszej wiadomości (klucz składający się z samych zer: 00 00 00 00 00 00 00, klucz składający się z samych jedynek: FF FF FF FF FF FF FF)
* Wadą algorytmu DES jest niewystarczająca 56-bitowa długość klucza, co sprawia, że jest bardzo podatny na ataki siłowe (ang. brute force).
* Zaletą algorytmu szyfrowania DES zdecydowanie jest nieskomplikowana implementacja na własne potrzeby.
* Algorytm DES świetnie nadaje się do szyfrowania plików na potrzeby prywatne, lecz szyfrowanie nim ważniejszych plików jest odradzane, ponieważ jest jednym z najlepiej przeanalizowanych algorytmów szyfrujących.