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 (wersja 17) w środowisku JetBrains IntelliJ. Użytkownik ma opcję wprowadzenia wartości heksadecymalnej klucza (którego rozmiar musi wynosić 64 bity), wpisania własnoręcznie tekstu jawnego poddanego później szyfrowaniu/deszyfrowaniu wedle wyboru, jak i wybrania opcji pobrania go z pliku.

1. **Algorytm DES**
2. Tekst jawny zamieniany jest na tablicę bajtów, następnie dzielony jest na bloki zawierające po 64 bity, w razie konieczności dopełniane znakiem „~” zapisanym bitowo (na początku pierwszego bloku), które są dodawane do listy.
3. Tworzone jest 16 podkluczy 48-bitowych za pomocą funkcji generateRoundKeys(key), gdzie key jest kluczem wprowadzonym przez użytkownika zapisanym w systemie binarnym:
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. Każda z nich przesuwana jest o 1 lub 2 bity (w zależności od wartości elementu tablicy iterationShift dla danego podklucza) w lewo, z wyjątkiem pierwszego bitu (lub 2 bitu), który jest umieszczany na końcu danej połówki.
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.
10. Powtarzamy kroki 2., 3., 4., 5., 6., jeszcze 15 razy w celu uzyskania pozostałych podkluczy 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, gdzie każda iteracja ma swój własny podklucz):
14. 32-bitową prawą stronę poddajemy permutacji rozszerzenia (Expansion Permutation), w celu uzyskania 48-bitowego bloku.
15. Rozszerzoną prawą połówkę poddajemy operacji XOR z podkluczem 48-bitowym (odpowiednim dla rundy).
16. Dokonujemy permutacji z tablicą S-Box (S-Box Substitution) zmienioną stroną prawą, w ten sposób uzyskujemy ponownie blok 32-bitowy.

* 48 bitowy blok prawej strony dzielimy na 8 bloków po 4 bity (dla każdego bloku istnieje odpowiadająca mu tablica w tablicy S-Box).
* Bity położone na zewnątrz (bity skrajne np. 011001) po połączeniu i przeliczeniu na postać dziesiętną będą wskazywać wiersz tablicy S-Box, a bity wewnętrzne (np. 011001) po przeliczeniu na wartość dziesiętną będą wskazywać kolumnę tablicy S-Box.
* Na podstawie operacji (numer\_wiersza · 16) + numer\_kolumny określany jest, który element z tablicy SBox ma być przeliczony na postać binarną (w razie potrzeby dopełniony do wartości 4-bitowej) i wstawiony w odpowiednie miejsce wyjściowego bloku.

1. Następuje permutacja 32-bitowej strony prawej z tablicą P (P-Box Permutation).
2. Dokonywana jest operacja XOR strony lewej ze zmienioną przez poprzednie kroki stroną prawą.
3. Otrzymany ciąg bitów jest zapisywany w miejscu prawej strony, a niezmieniona postać prawej strony (z początku danej rundy) jest zapisywana w miejscu lewej strony.
4. Obie strony łączone są w jeden 64-bitowy blok i poddawane permutacji z tablicą reversedIP (Final Permutation).
5. Zaszyfrowane 64-bitowe bloki zamieniamy na tablicę bajtów, które następnie konwertujemy do zapisu szesnastkowego i w rezultacie otrzymujemy wyjściowy zaszyfrowany tekst.

Deszyfrowanie odbywa się w sposób analogiczny, główna różnica polega na odwróceniu kolejności podkluczy dla kolejnych rund oraz usunięciu ewentualnego dopełnienia na początku pierwszego bloku. Odszyfrowana tablica bajtów zamieniana jest na tekst jawny zgodny ze standardem UTF-8.

**Wnioski końcowe**

* Wadą algorytmu szyfrowania DES jest możliwość wystąpienia tzw. kluczy słabych, czyli kluczy przy których wytworzone dla kolejnych rund szyfrowania podklucze będą takie same co ułatwi rozszyfrowanie naszej wiadomości (np. klucz składający się z samych zer: 00 00 00 00 00 00 00).
* Wadą algorytmu DES jest niewystarczająca 56-bitowa długość klucza, co sprawia, że jest bardzo podatny na ataki siłowe (ang. brute force) – ręczne sprawdzenie wszystkich możliwości.
* Zdecydowaną zaletą algorytmu szyfrowania DES jest nieskomplikowana implementacja oraz szybkie działanie.
* Algorytm DES nadaje się świetnie do szyfrowania plików na potrzeby prywatne, lecz szyfrowanie nim ważniejszych plików jest odradzane, ponieważ jest jednym z najlepiej przeanalizowanych algorytmów szyfrujących.