Bezpečnost v informačních technologiích

DES

(Semestrální práce)

Štěpán Martínek
smartine@students.zcu.cz
A15B0087P

Zadání

Implementujte šifru DES v prgramovacím jazyce C++.

Analýza

Algoritmus DES je obecně známy a jeho implementaci lze najít ve většině programovacích jazyků. Většina jeho implementací je ovšem změť příkazů nahuštěných do jedné funkce se spoustou magických čísel a bitových operací.

Návrh řešení

Pro řešení využiji moderní C++14. Mým cílem bude udělat jednoduchý a dobře čitelný (pro člověka znalého program bez využití magických čísel (s výjimkou transmutačních maticí). Program vytvořím podle veřejně známého postupu krok po kroku.

Popis řešení

Místo pracování s 64 bitovými integery, které využívají takřka všechny řešení jsem se rozhodl využít bitsety. Bitset je ADT která reprezentuje pole bitů určité velikosti, má přetížené bitové operace a možnost nastavovat jednotlivé bity pole pomocí funkce **set()**. Jelikož bitsety potřebují při vytvoření definovanou velikost a DES pracuje s různě velkými bloky, bylo potřeba po pěkný krátký kód využít templaty. Konkrétně templáty velikost, ty umožnují zároveň pracovat s danou velikostí tedy například tato funkce:

```
template<size_t T>
std::tuple<size_t, size_t> constexpr divide(std::bitset<T> const&
orig)
{
    std::bitset<T> divider(uint64(-1) >> T / 2);
    size_t rightKey = (orig & divider).to_ullong();
    size_t leftKey = ((orig >> (T / 2)) & divider).to_ullong();
    return { leftKey, rightKey };
}
```

vezme vstupní argument bitset velikosti T. Vytvoří si nový dočasný bitset velikosti T plný jedniček a shiftne ho o T/2 do pravá. Máme tedy bitset jehož levá půlka je plná nul a pravá půlka plná jedniček.následné bitovým & získáme pravou stranu a posunem originálního bitsetu a půlku doprava a opet bitovým & si dividerem levou stranu. Ty vracíme obě touplem. (Přesto že je vracíme jako size_t tak je přiřazuji do bitsetů které se z size_t vytvoří.

Programátorská dokumentace

Program lze nalézt na mém githubu: https://github.com/Kioshi/DES

Pro úspěšnou kompilaci je potřeba **CMake** verze 3.8+ a kompilátor podporující minimálně standart C++14.

Windows
Pro kompilaci pod systémem Windows vygenerujte přes **CMake** (GUI) projekt pro **Visual Studio (15+)**, ten následně otevřete a můžete program zkompilovat.

Pro debuggování je ještě potřeba nastavit projekt **DES** jako spouštěný projekt pomocí kontextového menu které se objeví po kliknutí na něj pravým myšítkem.

Linux
Vlezte do složky, kde chcete kompilovat projekt a příkazem „cmake <cesta ke root adresáři zdrojových souboru>" vygenerujte **Makefile.**

Nnásledně příkazem make[1] zkompilujte projekt.

Uživatelská dokumentace

Program DES.exe je konzolová aplikace. Která přijímá dva argumenty – klíč a text k zašifrování.

Použití:
DES.exe <klíč> <text k zašifrování>

Program vypíše zašifrovaný text jako hexadecimální string a následně ho rozšifruje a vypíše zpět v původní podobě.

Velikost klíče by měla být 64bitů, tedy 8 znaků.

Uživatelská dokumentace

[1] Aplikace byla vyvýjena pro Windows proto je možné, že bude potřeba do CMaku doplnit atribut pro g++/clang na využití daného c++ standartu.

Závěr

Podařilo se mi úspěšně naimplementovat DES pomocí moderního C++. První implementace podle návodu krok po kroku ovšem vůbec nefungovala. Musel jsem se proto uchýlit k tomu, že jsem šel krok po kroku s jiným funkčním programem a porovnával moje bitsety s cizími a upravoval program aby dával stejné výsledky. Většinou se jednalo pouze o reverzy bitsetů. K tomuto krokování se bitsety vyložene hodí jelikož je možně je přímo vypsat na rozdíl od reprezentace Integerem která je potřeba ručně krok po kroku vypsat bit po bitu.