

ZÁPADOČESKÁ UNIVERZITA V PLZNI

Fakulta aplikovaných věd

KIV/BIT

Implementace SHA-1

Autor: Lukáš Haringer, A13B0306P

Akademický rok: 2015/2016

Počet hodin: 15

Obsah

| | |
|-------------------------------|---|
| Zadání | 3 |
| Analýza problému..... | 3 |
| Hashovací funkce..... | 3 |
| SHA-1 | 3 |
| Bezpečnost | 3 |
| Návrh řešení | 4 |
| Popis řešení | 5 |
| Třídy aplikace..... | 5 |
| Metody třídy SHA1 | 5 |
| Uživatelská dokumentace | 6 |
| Spuštění aplikace..... | 6 |
| Ovládání aplikace | 6 |
| Závěr | 7 |

Zadání

Zadáním semestrální práce je implementovat hashovací funkci SHA-1 ve vhodném programovacím jazyce.

Analýza problému

Hashovací funkce

Hashovací funkce je funkce, která z libovolného vstupu dat vytváří výstup fixní délky, který je označován jako hash, otisk, miniatura a podobně (anglicky fingerprint). Jeho hlavní vlastností je, že malá změna na vstupu vede k velké změně na výstupu, tj. k vytvoření zásadně odlišného otisku.

SHA-1

SHA navrhla organizace NSA (Národní bezpečnostní agentura v USA) a vydal NIST (Národní institut pro standardy v USA) jako americký federální standard. SHA-1 vytvoří obraz zprávy dlouhý 160 bitů. SHA-1 hash je často interpretován jako čtyřiceti místné hexadecimální číslo.

Bezpečnost

Hashovací algoritmy jsou „bezpečné“, pokud je velmi obtížné (tj. se současnými prostředky prakticky nemožné):

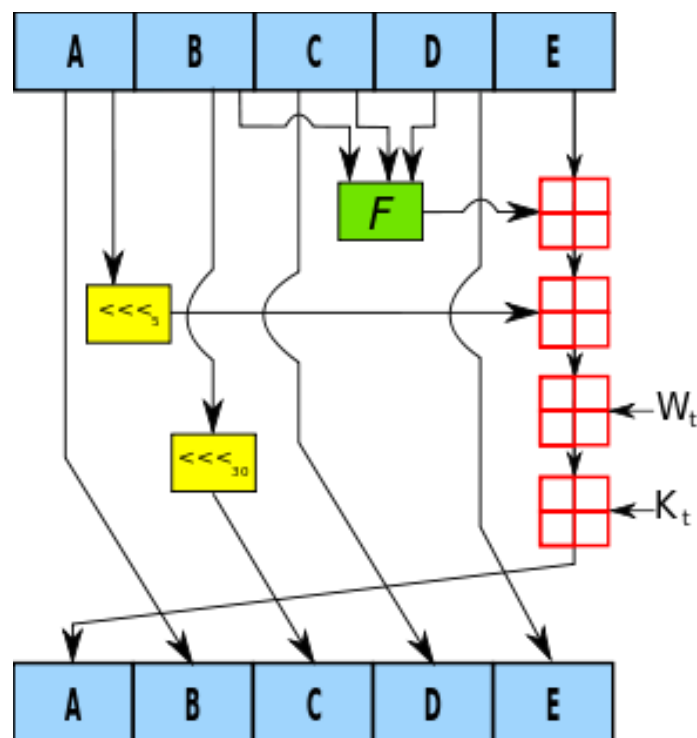
1. Najít zprávu, která odpovídá svému otisku
2. Najít dvě rozdílné zprávy, které mají stejný otisk

SHA-1 algoritmus již není považován za dostatečně bezpečný. Již v roce 2005 kryptoanalytici našli útok, který naznačoval, že další používání tohoto algoritmu nebude bezpečné. Od roku 2010 mnoho organizací doporučilo jeho nahrazení algoritmy SHA-2 nebo SHA-3. Microsoft, Google and Mozilla oznámili, že jejich prohlížeče přestanou uznávat SHA-1 SSL certifikáty od roku 2017.

Návrh řešení

1. SHA-1 algoritmus nejdříve doplní zprávu tak, že přidá bit 0x80 a délku zprávy v bitech jako 64-bit big-endian integer a poté zprávu doplní nulami tak aby byla její délka dělitelná 512 bity.
2. Poté je zpráva rozdělena na bloky po 512 bitech.
3. Jednotlivé bloky jsou poté zpracovávány tak, že jsou rozděleny na 16 32-bit big-endian slov
4. Z původních 16 slov W je poté za pomoci následujícího algoritmu vytvořeno 80 slov.

$$w[i] = (w[i-3] \text{ xor } w[i-8] \text{ xor } w[i-14] \text{ xor } w[i-16]) \text{ leftrotate } 1$$
5. Každé slovo je poté zpracováno dle následujícího schématu.



A, B, C, D a E jsou 32-bitová slova vznikající úpravami konstant H.

F – je nelineární funkce.

<<< - levá bitová rotace o n míst

W_t je slovo dané iterace t

K_t je konstanta dané iterace t

\boxplus součet modulo 2^{32}

6. Po dokončení všech iterací dostaneme výsledný hash daného řetězce.

Popis řešení

Semestrální práce je implementována v jazyce Java.

Třídy aplikace

Aplikace se skládá ze dvou tříd:

1. **Main** – Hlavní třída aplikace. Stará se o vytvoření uživatelského rozhraní.
2. **SHA1** – Třída, která se stará o hashování vstupu.

Metody třídy SHA1

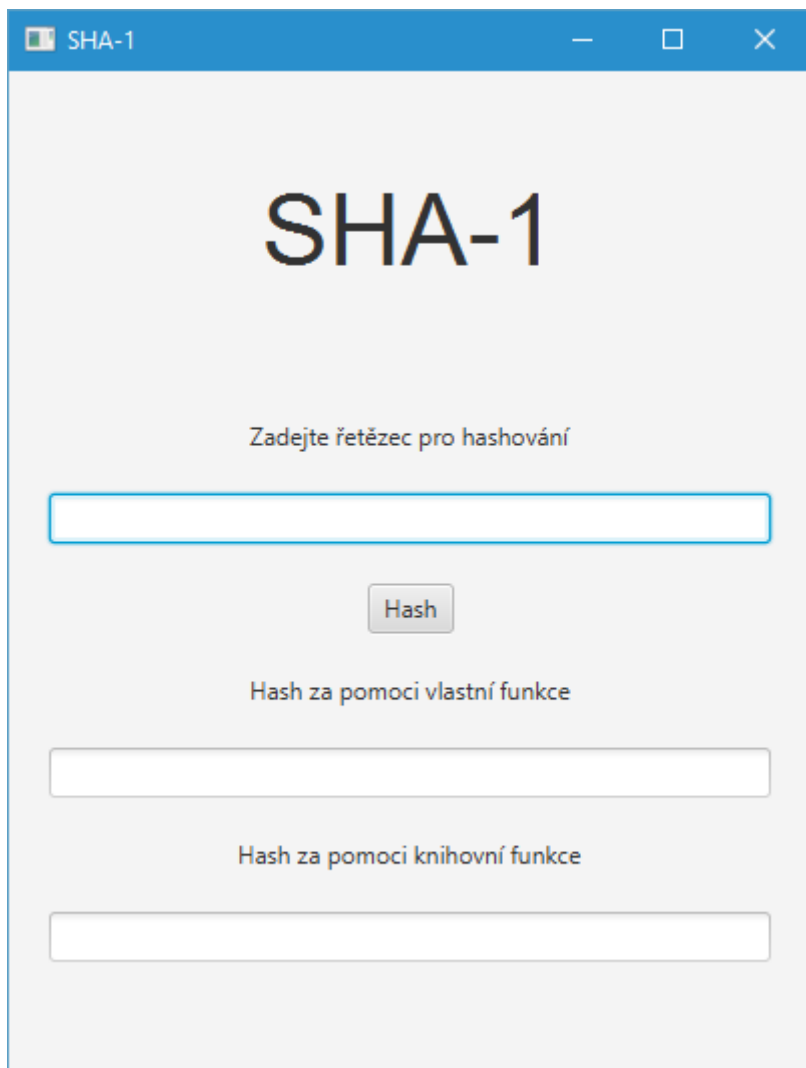
1. **String hashItLibrary(byte[] dataIn)** – Metoda, která vytvoří ze vstupních dat hash SHA-1 za pomoci knihovní funkce. (Slouží pro porovnání správnosti našeho hashe)
2. **String hashIt(byte[] dataIn)** – Metoda, která vytvoří hash SHA-1 ze vstupních dat. Požadovaný hash vrací jako String.
3. **byte[] padTheMessage(byte[] data)** – Metoda, která se stará o doplnění zprávy a požadované informace a doplnění délky tak aby byla délka zprávy dělitelná 512.
4. **void processTheBlock(byte[] block, int H[])** – Metoda, která zajišťuje zpracování bloku zprávy. Rozloží blok na 80 slov a provede dané iterace.
5. **int rotateLeft(int value, int bits)** – Metoda pro bitový posun do leva o bits míst.
6. **String intArrayToHexStr(int[] arr)** – Metoda pro převod pole integerů na řetězec.

Uživatelská dokumentace

Spuštění aplikace

Aplikace se spouští dvojklikem na soubor BIT-SHA1.jar. Ke spuštění aplikace je potřeba mít nainstalováno JRE 1.8.

Ovládání aplikace



SHA-1

Zadejte řetězec pro hashování

Hash

Hash za pomoci vlastní funkce

Hash za pomoci knihovní funkce

Do pole **zadejte řetězec pro hashování** zadáme požadovaný řetězec, pro nějž chceme získat hash. Po stisknutí tlačítka hash se spočítá požadovaný hash za pomoci knihovní a vlastní funkce a zobrazí se v příslušných polích.

Závěr

Zadání semestrální práce jsem splnil v plném rozsahu. Aplikace vytváří SHA-1 hashe pro zadané vstupní řetězce. Při práci na této semestrální práci jsem získal znalost o fungování hashovacích funkcí, které jsem dříve používal pouze pasivně.