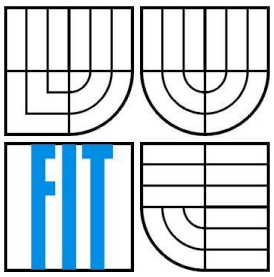
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ



KRY PROJEKT 2

Vladimír Veselý – xvesel38

OBSAH

1 ÚVOD

1.1 ABSTRAKT

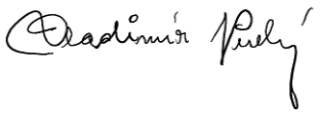
Tento dokument analyzuje a řeší projekt číslo 2 do předmětu Kryptografie pro letní semestr roku 2009, specifikovaný zadáním na adrese:

http://www.buslab.org/index.php/content/view/134081/64/

1.2 PROHLÁŠENÍ

Tímto čestně prohlašuji, že práce ani žádná její část není plagiátem a že byla vypracována samostatně v souladu s dobrými mravy a přísnou morální etikou kladenou na bedra studenta imatrikulačním slibem. Svým vlastnoručním podpisem stvrzuji pravdivost tohoto prohlášení.

5.5.2009

X 

Vladimír Veselý

2 IMPLEMENTACE

2.1 PLATFORMA

Generické řešení pro digital watermarking nad 24-bitovými obrázky BMP (které jsou v barevném formátu RGB, kde každá složka je právě 8-bitová) bylo implementováno v jazyce Java 1.6u13.

Projekt byl vyvinut v IDE NetBeans 6.5.1, pro který jsou určeny i spustitelné soubory.

Samotná implementace je rozprostřena do několika tříd:

- **KRYApp** – aplikační logika, obsahuje zejména konverzní funkce (*intToByteArray*, *byteArrayToInt*, *byteArrayToBitset*, *byteToBitset*, *bitsetToByteArray*, *bitsetToByte*);
- **KRYView** – GUI aplikace;
- **Vlakno** – stará se o řízení procesu zašifrování textu do obrázku, paralerně spouští vláknové potomky a synchronizuje se nad jejich výsledkem;
- **VlaknoData** – samostatné vlákno, které pozměňuje obsah jediného pixelu, přičemž koordináty tohoto pixelu v rámci rastru jsou vstupním argumentem konstruktoru;
- **JPictureGUI** – prekompilovaná komponenta pro zobrazování obrázků stretchlých do rastru o specifikovaných velikostech;
- **KRYAboutBox** – okénkový formulář „O programu“ obsahující rádobyvtipné sdělení;

2.2 ALGORITMUS

Algoritmus samotný nejprve rozloží vstupní text na sekvenci bytů, tuto pak dále na sekvenci bitů. Poté danou sekvenci postupně ukládá do jednotlivých pixelů obrázku tím, že modifikuje nejméně významný bit modré části barevného modelu obrázku.

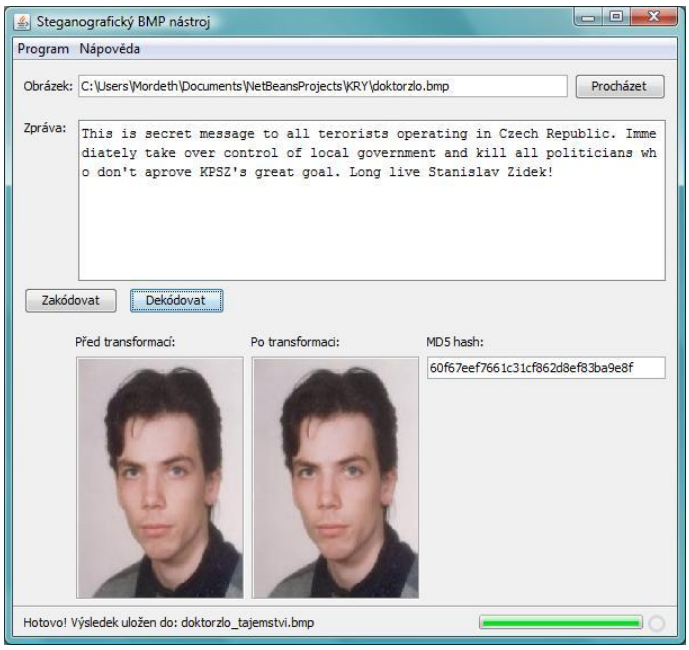
Před tímto krokem však ještě provede kontrolu, zda je hostitelský dostatečně velký, aby pojal zašifrovávaný text. Protože každý pixel obrázku kóduje 1 bit řetězce zprávy rozložené na bity a s přihlédnutím aspoň k dvojné redundanci (výsledný text obsahuje skrytou zprávu nejméně 2×), je nutné, aby byla tedy splněna následující podmínka:

$$Picture.width * Picture.height \geq 2 \cdot Message.bitSize$$

2.3 SPUŠTĚNÍ

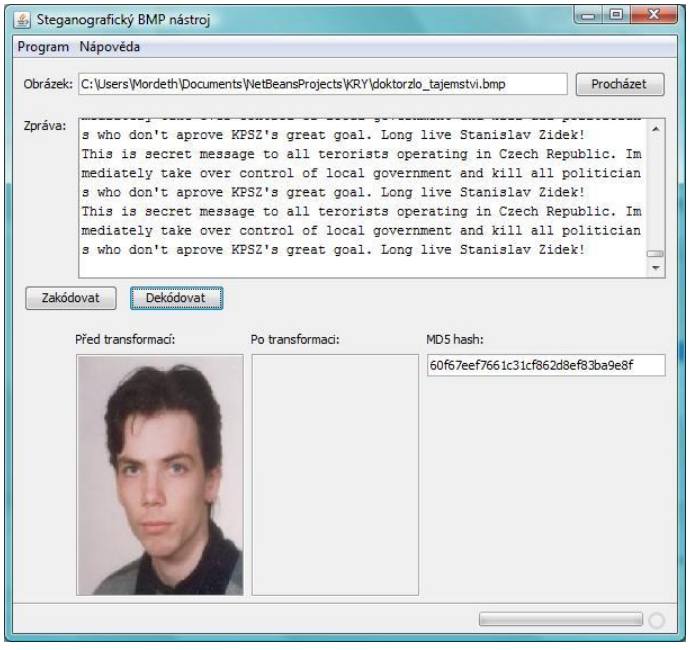
Program lze zkompilovat ze zdrojových kódů anebo spustit připravenou dávku „run.bat“ či „run.sh“ (v závislosti na operačního systému), která pak nastartuje projekt z předkompilovaného binárního souboru KRY.jar.

2.4 OVLÁDÁNÍ ZAKÓDOVÁNÍ



Nejprve vybereme hostitelský obrázek pomocí tlačítka „Procházet“; Poté vložíme text, který chceme zašifrovat;
Klikneme na tlačítko „Zakódovat“ čímž se zahájí proces úpravy, jehož stav zpracování je pozorovatelný na progressbaru;
Po skončení je výsledek uložen do souboru, který má shodný název jako vstupní obrázek, jen s tím, že je k němu přilepena přípona „_tajemstvi“.
Posledním krokem běhu programu je vygenerování kontrolního MD5 hashe pro verifikaci ze strany příjemce tajné zprávy.

2.5 OVLÁDÁNÍ DEKÓDOVÁNÍ



Postup je analogický. Pomocí tlačítka „Procházet“ načteme obrázek se skrytou zprávou a poté klikneme na tlačítko „Dekódovat“.

Výsledek dešifrování je zobrazen do textového pole a je vygenerován kontrolní hash pro ověření, že zpráva neobsahuje chyby.