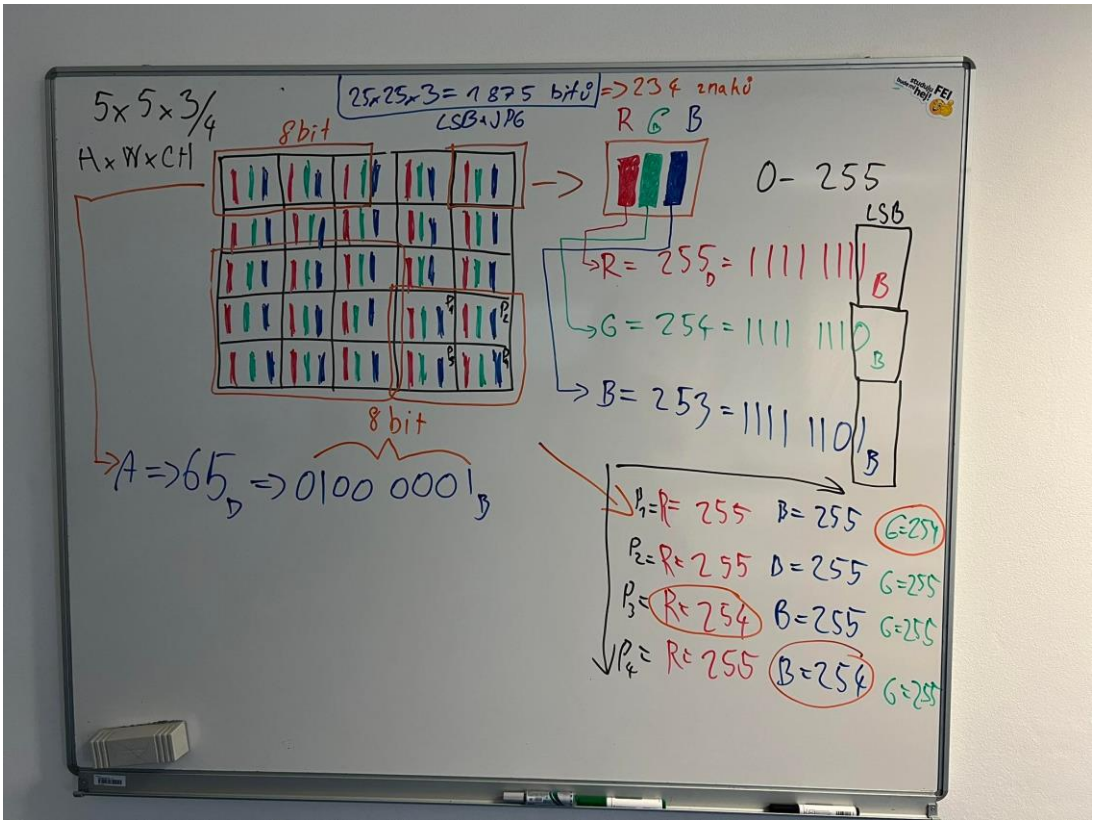
Jméno a příjmení: **Adam Šárek**
Osobní číslo: **SAR0083**
Datum: **16.10.2022**

Forenzní analýza - Protokol (3)

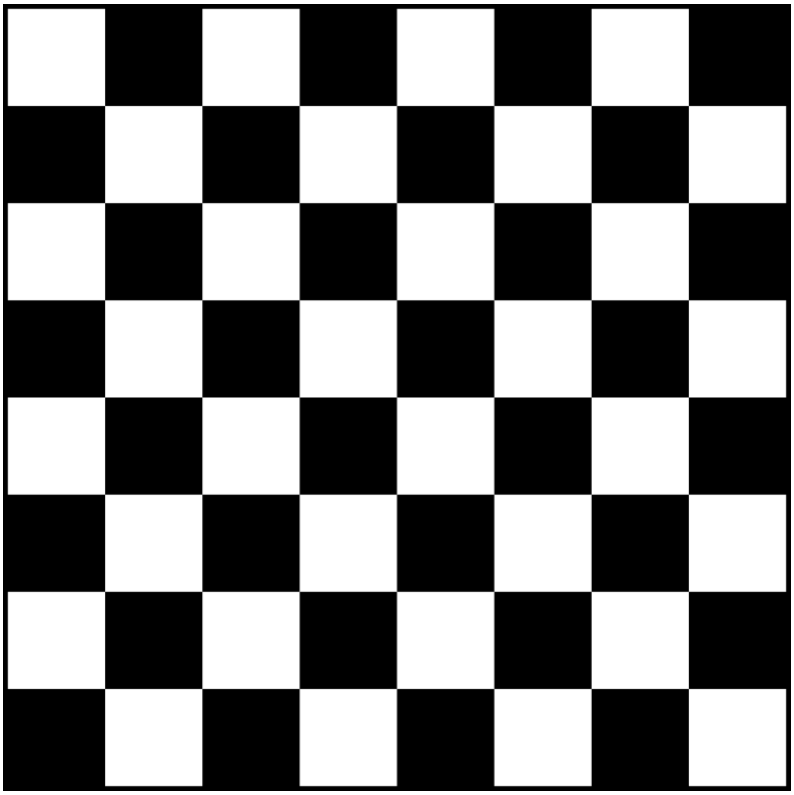
Steganografie

1. **Seznamte se s problematikou steganografie.** ***(POZOR! Steganografie není pouze o ukrývání informací do obrázku, jak je často mylně označováno.)***
2. **Seznamte se s problematikou skrývání informací (text / soubor) do obrázku.**
3. **Seznamte se s problematikou detekce steganografie.**



4. Na základě informaci o steganografii, vytvořte program, který bude obsahovat tyto funkce:
    a. Funkce pro ukládání textu zadaného uživatelem v programu. Program musí provést kontrolu, zda se daný text do obrázku vleze. **(1 b)**
    b. Funkce pro ukládání libovolného souboru do obrázku, včetně kontroly zda se obrázek vleze. **(1 b)**
        i. **BONUS!** Pokud se soubor do obrázku nevleze, nabídněte uživateli možnost zvětšení obrázku pro zvýšení kapacity. **(1b)**
        ii. Doporučení! Zvažte implementaci hlavičky, která bude obsahovat informace o skrytém souboru / textu. Typicky by hlavička mohla obsahovat tyto informace:
            1. Typ uložené informace (Soubor / Text) = 1 bit
            2. Způsob uložení informace = 2-8 bitů (0 = každý pixel, 1 = každý sudý pixel, 2 = každý lichý pixel, 3 = každý pixel na okrajích obrázku).
            3. Název uloženého souboru = 64 * 8 bitů (max. 64 znaků)
            4. Pozice prvního bitu s informaci = 32 bitů
            5. Pozice posledního bitu s informaci = 32 bitů
        iii. Každopádně návrh a implementace hlavičky je na vás.

c. Funkce pro získání uložených dat (text / souboru) z obrázku zpátky. **(1 b)**
d. Funkce pro detekci steganografie v obrázku (ne jen obrázku vygenerovaného vaším programem). Pro usnadnění implementujte detekci jednoduchou a jako testovací obrázek použijte šachovnici. **(1 b)**
    i. **BONUS**! Implementujte a popište sofistikovanou detekci steganografie, které jdou použít i na detekce ve fotkách. **(2 b)**



5. Navrhněte způsob, kam jinde ukrývat informace pomoci steganografie. (Mimo obrázky a video!). Tento návrh teoreticky důkladně popište. **(2 b)** -> Příklad z minulých let: PDF soubor, zvuková nahrávky, webová stránka, hra, .exe soubor
    a. **BONUS**! Tento teoretický návrh implementujte, alespoň v minimalistické formě (bez kontrol), pouze s funkčnosti uložit a zpětně vyčíst. **(2 b)**

Otázky

1) Popište jakou funkci může vykonávat steganografie v oblasti autorského práva. Uveďte minimálně 5 příkladů, včetně toho jak je lze provést. **(1 b)**
2) Detailně popište rozdíly mezi steganografii a kryptografii. **(1 b)**
3) **BONUS!** Najděte a detailně popište popište nekonvenční (ne obrázky a video) nástroj pro digitální steganografii. **(1 b)**

Vypracování

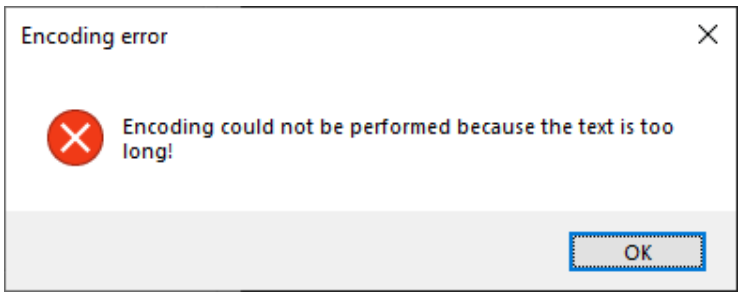
Ve vypracování je několikrát zmíněno vytvořené řešení v podobě desktopové aplikace jejíž zdrojový kód je součástí odevzdaného řešení.

Ukládání textu a souboru, kontrola velikosti obsahu

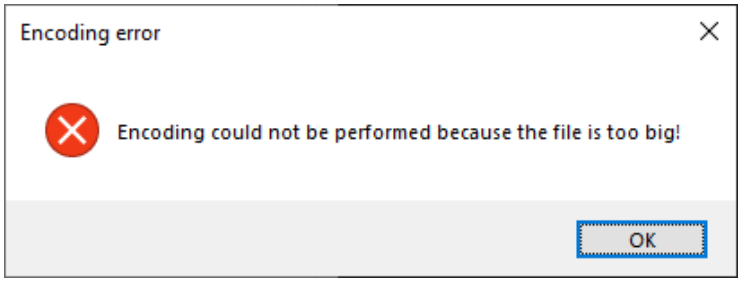
V rámci vytvořeného řešení je ukládání textu i souboru řešeno jednotně skrze metodu *BitmapData.Encode()*, která na začátku kontroluje, zda je daná velikost obsahu menší než celková dostupná velikost pro obsah. Celková velikost pro obsah je počet všech dostupných bitů (šířka x výška x 3) od kterého je odečtený počet bitů pro hlavičku.

Ukládání textu se od ukládání souboru liší tím, že je prováděno skrze metodu *BitmapData.EncodeText()*, která ovšem pouze nastaví formát obsahu na „txt“ a převede text to bytové podoby (tedy stejné v jaké jsou načítány soubory). Dále se pak již s textem pracuje stejně jako v případě souboru, a to skrze již zmíněnou metodu *BitmapData.Encode()*.

V případě, že je velikost v pořádku, tak se soubor uloží na předem zvolené místo v paměti. V opačném případě se pak zobrazí vhodná varianta okna viditelného na dvou obrázcích níže.



Obr. 1 - Chybová hláška pro příliš dlouhý text



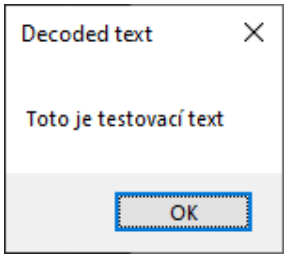
Obr. 2 - Chybová hláška pro příliš velký soubor

Implementace hlavičky

Hlavička v rámci řešení obsahuje příponu souboru (v případě textu „txt“) a počet bitů použitých pro samotný obsah. Hlavička obsahuje 48 bitů pro zaznačení přípony souboru a proměnný počet bitů pro zaznačení počtu využitých bitů pro obsah. Proměnný počet je zde z toho důvodu, že pro různé velikosti obrázku se také různí počet použitelných bitů, a tedy je vhodné mít i proměnný počet bitů pro zápis tohoto počtu. Informace jsou ukládány do sloupců pixel po pixelu na nejméně významných bitech RGB kanálů barev. Pozice prvního bitu je dána koncem hlavičky a pozice posledního bitu je dána koncem hlavičky + počtem použitých bitů. Název uloženého souboru ukládán není.

Získání uložených dat z obrázku

Pro získání uložených dat je použita funkce *BitmapData.Decode()*, která nejprve zkontroluje, zda byla steganografie v obrázku detekována. Pokud ano, tak z již načtených dat v podobě bytového pole, RGB kanálů v řádcích, sloupcích a pixelech, získává nejprve hlavičku a po zjištění počtu použitých bitů pro obsah pak načítá i samotný obsah. Tento obsah je poté možné buď zobrazit, pokud se jedná o text (formát „txt“), nebo jej uložit do souboru. V aplikaci pak zobrazení textu vypadá podobně jako na obrázku níže.

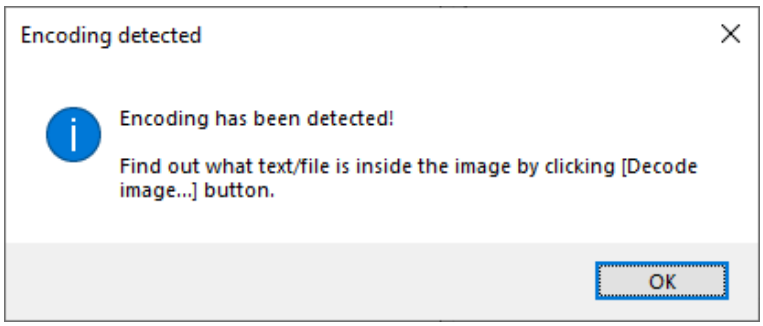


Obr. 3 - Hláška pro zobrazení textového obsahu

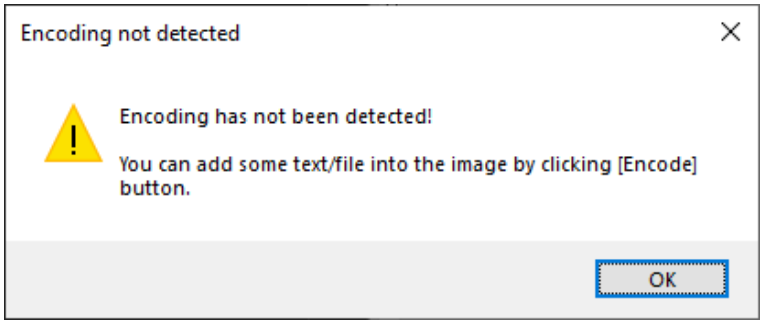
Detekce steganografie

Steganografie je v obrázku detekována funkcí RawBitmapData.IsEncodingDetected(), která kontroluje vždy čtverec 4 sousedních pixelů a porovnává jejich barevné hodnoty. Aby byla steganografie detekována, pak v rámci libovolného čtverce pixelů, musí mít všechny 4 pixely shodné všechny bity barvy až na nejméně významný bit. Ten se pak musí lišit právě u jednoho pixelu. Pokud toto platí u všech 3 barevných kanálů v této čtveřici pixelů pak je v daném obrázku detekována steganografie.

V případě, že bychom se pokoušeli provést tuto kontrolu u libovolného importovaného obrázku v aplikaci, pak by to vedlo k jedné z následujících možností.



Obr. 4 - Hláška pro detekovanou steganografii



Obr. 5 - Hláška pro nedetekovanou steganografii

Závěr

Steganografie v oblasti autorského práva

1. **Vložení informace o copyrightu do obrázku** může sloužit k usvědčení z nedovoleného použití daného obrázku bez svolení autora. Funguje tedy jako digitální vodoznak a plní funkci podpisu majitele autorských práv. V praxi je možné jej provést podobně jako to již bylo zmíněno v předchozí kapitole.
2. **Označkování studiových verzí písní** je využito při vytváření hudebních alb, a to pro případné úniky studiových nahrávek. V případě, že by k takovému úniku došlo, pak by dle značky bylo možné snadno dohledat osobu, která tento únik provedla.
3. **Skrytí obrázku či souboru do videa** je metoda, kterou je možné označit např. film či jiné audiovizuální soubory, které mohou být potenciálně šířeny v rozporu s autorským právem. Lidské oko pak není schopno drobné změny zachytit, jelikož ve videu je zpravidla 24 či více snímku za sekundu.
4. **Neviditelný inkoust** je forma neviditelného písma, které může být součástí dopisů, významných listin či knih. Dnes se tato metoda používá např. u výroby bankovek.
5. **Označení elektronické pošty** pomáhá odhalit spam či falšované zprávy, které mohou obsahovat prvky původní pošty. Steganografie zde může být buď přímo v daném textu zprávy či skrytá v html kódu.

Rozdíly mezi steganografií a kryptografií

Steganografie se používá pro skrytou komunikaci, zatímco kryptografie se používá pro ochranu dat. Steganografie nemění strukturu dat, což vede k tomu, že je těžší ji odhalit. Kryptografie mění strukturu dat tím, že je šifruje. Zašifrovaná data je pak těžké prolomit, jelikož je k tomu potřeba buď klíč či hrubá výpočetní síla. Kryptografie se používá pouze pro text, zatímco steganografie navíc také pro obrázky, videa či audio. Kryptografie se používá častěji než steganografie.

Nekonvenční nástroj pro digitální steganografii

Steghide je open-source nástroj, který pomocí několika příkazů v příkazovém řádku umožňuje skrýt soubor do různých formátů obrázků a audia. Podporuje formáty JPEG, BMP, WAV a AU. Umožňuje také komprimovat a šifrovat vkládaná data a dále pak také extrahovat vložená data či detekovat jejich integritu s pomocí předem vloženého kontrolního součtu. Vzhledem k tomu, že byl tento nástroj vyvinut již před delší dobou, tak je možné jej spustit pouze na 32bitových verzích operačního systému Windows.