VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ
DEPARTMENT OF COMPUTER SYSTEMS

DIGITÁLNÍ TEXTOVÁ STEGANOGRAFIE

DIGITAL TEXT STEGANOGRAPHY

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **PETR POUČ**
AUTHOR

VEDOUCÍ PRÁCE **Ing. JOSEF STRNADEL, Ph.D.**
SUPERVISOR

BRNO 2022

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačových systémů (UPSY) Akademický rok 2021/2022

Zadání bakalářské práce

Student: **Pouč Petr**
Program: Informační technologie
Název: **Digitální textová steganografie**
Digital Text Steganography
Kategorie: Bezpečnost

Zadání:

1. Vytvořte přehled metod z oblasti digitální steganografie, detailněji pak přehled z podoblasti steganografie pro skrývání informace v textových datech (dále jen "textová steganografie") a jejich vlastností. Shrňte současný stav a trendy v této podoblasti.
2. Zvolte typ skrývané informace (textová, obrazová apod.) a její vlastnosti. Na základě existující či vlastní analýzy způsobů ukládání textových dat a typu skrývané informace zvolte vhodné způsoby pro ukládání dat a vhodné metody pro textovou steganografii.
3. V souladu s bodem 2 připravte vhodnou sadu dat pro ověřování vlastností zvolených steganografických metod a navrhněte mechanismus vyhodnocování jejich vlastností na základě těchto dat.
4. Implementujte několik existujících metod textové steganografie; zvažte jejich modifikace. Navrhněte a implementujte alespoň jednu vlastní metodu.
5. Ověřte funkčnost implementovaných metod; vyhodnoťte jejich vlastnosti a porovnejte je jak navzájem, tak s daty z vybraných publikací.
6. Shrňte dosažené výsledky, diskutujte možné směry využití a rozvoje předloženého řešení.

Literatura:

- Dle pokynů vedoucího.

Pro udělení zápočtu za první semestr je požadováno:

- Splnění bodů 1 až 3 zadání.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Strnadel Josef, Ing., Ph.D.**
Vedoucí ústavu: Sekanina Lukáš, prof. Ing., Ph.D.
Datum zadání: 1. listopadu 2021
Datum odevzdání: 11. května 2022
Datum schválení: 29. října 2021

Abstrakt

Tato bakalářská práce se zaměřuje na rozbor vědní disciplíny zvaná steganografie, konkrétně digitální textovou steganografii.

Cílem této práce bylo se seznámit s tímto oborem a podrobně prozkoumat metody z oblasti digitální steganografie. Součástí bakalářské práce bylo vybrat a implementovat vhodné metody pro skrývání informací v textových datech.

Metody jež jsem si zvolil a naimplementoval jsem na závěr podrobil řadě automatizovaných testů a podrobné analýze. Provedeným výzkumem jsem zjistil, že zprávy ukryté těmito metodami dokáží být velmi špatně postřehnutelné, a jedná se tak o velmi spolehlivý způsob posílání důvěrných dat. Vytvořené řešení bylo zpracováno do přehledné tabulky, ve které lze vidět, že nepostřehnutelnost některých metod překonala hranici až 95 % a úspěšnost ukrytí krátkých zpráv u některých metod dosahovala téměř 100 %.

Výsledky této práce umožňují zvolit vhodnou metodu pro nejspolehlivější vložení informace do textového souboru a zamezit tak nechtěnému dešifrování skryté zprávy.

Abstract

This bachelor's thesis aims to identify the characteristics of a field of science called steganography, especially digital text steganography.

The purpose of this study is to become acquainted with this field and analyze some of the methods of digital text steganography in a detail. Another objective of this bachelor's thesis was to choose and implement suitable methods for information hiding in text data.

The chosen methods that were implemented were finally verified by automated tests and analyzed in detail. Based on the results of this research, it can be concluded that secret messages hidden by these methods can be very difficult to spot. The final statistics were processed into a comprehensible table, in which we can see, that perceptibility of some methods got over 95 % and success rate of hiding short messages reached almost 100 %.

Based on the results of this study, we can choose a suitable method for the most reliable information hiding in the text file, so we can prevent unwanted decryption of a hidden message.

Klíčová slova

Steganografie, skrývaní informací, změna textového souboru, bezpečnost, lingvistika, tajná komunikace, důvěrná data

Keywords

Steganography, information hiding, text file moodification, security, linguistics, secret communication, confidental data

Citace

POUČ, Petr. *Digitální textová steganografie*. Brno, 2022. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Josef Strnadel, Ph.D.

Digitální textová steganografie

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana doktora Strnadela. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Petr Pouč
7. května 2022

Poděkování

Především bych chtěl poděkovat vedoucímu mé bakalářské práce doktoru Josefu Strnadelovi za veškerou odbornou pomoc, konzultace a užitečné rady k vylepšení této práce. Dále bych rád poděkoval své přítelkyni a rodině za jejich trpělivost a podporu.

Obsah

Kapitola 1

Úvod

Steganografie byla po dlouhou dobu oblíbeným způsobem utajené komunikace nejen pro armádu, ale i pro politické vůdce. V dnešní moderní době je naprostá většina informací přenášena přes internet, a proto jsou možnosti využití steganografie větší, než kdy dříve.

Skrývání různých typů informací, kombinování různých metod a nástrojů nám zajišťuje vysokou rozmanitost využití. Nyní máme možnost pomocí steganografie nepozorovaně doručit téměř jakoukoliv informaci. Můžeme do obrázku schovat plán na sestavení letadla, nebo do videa nepozorovaně vložit seznam hesel. Steganografie je také často využívána v oblasti ochrany autorských práv [24], skrytá informace v dokumentu snadno prokáže jeho legitimního vlastníka.

V současné době, kdy se technologie a její uživatelé závratným tempem zdokonalují, je důvodů k ochraně svých údajů větší než kdy dříve. Růst v používání internetu, chytrých telefonů a jiných moderních zařízení má velký dopad na náš každodenní život. Tyto nové technologie jsou velkým přínosem, umožňují nám zpracovávat informace nenákladným a široce dostupným způsobem, avšak tím roste i strach o únik těchto dat. [3]

Naštěstí existuje obor zvaný steganografie, který nám poskytuje mnoho steganografických metod, které dokáží ukrýt citlivé informace tak, aby běžného pozorovatele bez speciálních znalostí nenapadlo tyto data hledat, natož se k těmto informacím dostat. Cílem této práce bude vás s tímto nepříliš známým oborem seznámit a rozšířit vám obzor v oblasti skrývání digitálních dat.

Kapitola 2

Rešerše k tématu bakalářské práce

2.1 Základní pojmy

Ještě než stručně shrnu historii steganografie a popíšu odlišné způsoby jejího provedení, bylo by vhodné na začátek objasnit, co tento termín znamená. Steganografie je vědní obor, zabývající se utajenou komunikací. Původem tento pojem pochází z řečtiny a doslovně znamená „skryté psaní“ [12]. Cílem steganografie je nepozorovaně zašifrovat zprávu a bezpečně ji tak dopravit požadovanému adresátovi.

2.1.1 Rozdíl mezi steganografií a kryptografií

Steganografie spadá pod vědní obor nazývající se kryptografie. Hlavní rozdíl těchto dvou disciplín je v přístupu k ukrytí informace. Steganografie usiluje o ukrytí informace nepozorovaně tak, aby zpráva, obrázek, ani zvukový soubor nebudily žádné podezření. Zatímco kryptografie šifruje zprávu zcela otevřeně a výsledná podoba vypadá jako nesmyslná kupa znaků [12].

	Steganografie	Kryptografie
Cíl	Ukrytí zpráv bez šifrování obsahu	Zašifrovaní obsahu zprávy
Utajení	Skrytá zpráva není nijak patrná Text v pozorovateli nebudí žádné podezření	Je okamžitě patrné, že se jedná o zašifrovanou zprávu
Zabezpečení	Záleží na použité metodě	Záleží na diskrétnosti klíče
Útoky	Náročné odhalení/Náročná extrakce	Snadné odhalení/Náročná extrakce

Tabulka 2.1: Srovnání Steganografie a Kryptografie. [29]

2.1.2 Vznik steganografie

První použití steganografie je připisováno Řekům. Zdokumentováno bylo antickým historikem Herodotem, podle nějž steganografie zachránila Řecko před Xercem [12].

Demaratus, Řek žijící v Persii se rozhodl varovat Sparťany před Xerxesovou invazí do Řecka. Demaratus na dřevěné skládací destičky nanesl vrstvu vosku, pod kterou se skrývala skrytá zpráva [12].

2.2 Historie steganografie a její vývoj

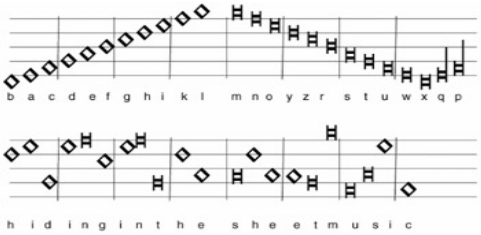
V této kapitole jsem jako primární zdroj používal knihu `Investigator's guide to steganography` [12]. Tento zdroj v této kapitole nadále nebudu uvádět. Uvedené budou pouze citace, které výtažky z této knihy doplňují.

Steganografie má dlouhou historii, její vznik je datován již od starověku (viz 2.1.2), kde byl tento způsob ukrytí zprávy poprvé použit formou dřevěného tabletu a vrstvy vosku. Postupem času se tato věda vyvíjela a zdokonalovala až do dnešní podoby.

V této kapitole vám představím hlavní milníky vývoje steganografie a její nejdůležitější průkopníky, kteří přispěli ke zdokonalení tohoto oboru.

2.2.1 Gaspar Schott

Tento německý vědec vytvořil dílo *Schola Steganographica*, kde poprvé popisuje steganografickou metodu, založenou na přiřazování písmen jednotlivým notám. Píseň by nebyla příjemná k poslechu, avšak nezkušenému člověku by se noty jevily zcela obyčejně.



Obrázek 2.1: Steganografická metoda Gaspara Schotta (převzato z [12]).

2.2.2 Johannes Trithemius

Spisovatel a kryptograf Johannes Trithemius je jedním ze zakladatelů moderní steganografie. Okolo roku 1500 napsal třídílné dílo *Steganographica*. Tato kniha popisuje rozsáhlý systém k ukrývání tajných zpráv do nevinně vypadajícího textu.

Popsal zde několik nových steganografických metod. Příkladem je metoda, která z cover textu[1] bere každé 2. slovo a z něj každé 2. písmeno. Z těchto znaků je následně složena tajná zpráva.

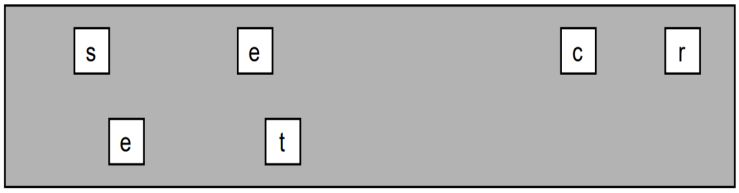
padiel a**p**o**r**s**y** mesarpon o**m**e**u**a**s** peludyn m**a**l**p**r**e**a**x**o

V následující zprávě je ukryta skrytá zpráva *prymus apex*. Další inovativní metodou v knize *Steganographia* byla tzv. šifra *Ave Maria*. Tato metoda obsahuje několik tabulek, které přiřazují každému písmenu nějaké slovo. K napsání tajné zprávy autor textu nahrazuje písmena vybranými slovy, jenž jsou obsaženy v tabulkách.

[1] Cover text - text do kterého je vložena tajná zpráva.

2.2.3 Girolamo Cardano

Tento vynikající matematik přišel se způsobem ukrývání zpráv, s kterým jsme dosud neměli možnost se setkat, tzv. grille system. Každý příjemce má papír s několika otvory. Pokud grille[2] překryjeme přes obyčejně vypadající text, díry nám odkryjí tajnou zprávu.



Obrázek 2.2: Ukázka vytvoření tajné zprávy pomocí grille (převzato z [12]).

2.2.4 Bishop John Wilkins

Bishop John Wilkins přišel s nápadem pro tajné psaní pomocí materiálů svítících ve tmě.

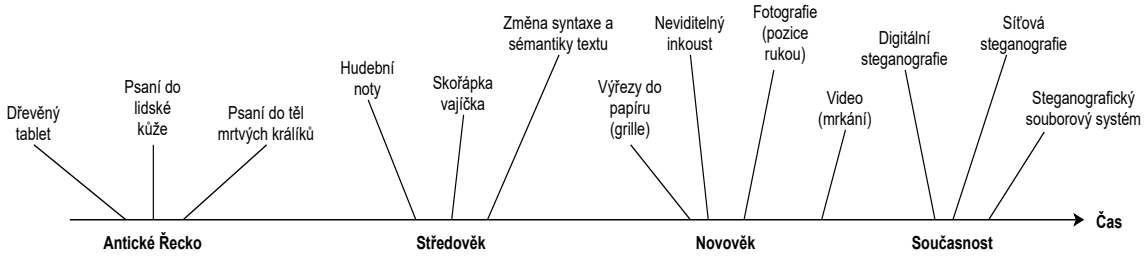
2.2.5 The USS Pueblo

Roku 1968 byla při misi v Severní Koreji zajata americká loď s posádkou. Po několika měsících byla zveřejněna fotka posádky, která měla dokazovat spolupráci s Koreou. Tato fotka však obsahovala tajnou steganografickou zprávu. Pozice rukou členů posádky představovaly tajnou zprávu *snow job*. Tato fráze znamená utajení skutečného motivu ve snaze někoho přesvědčit.

2.2.6 Válka ve Vietnamu

Dalším příkladem využití steganografie je případ amerického velitele Jeremiah Dentona, který byl zajat během války ve Vietnamu. Byl donucen veřejně vystoupit v televizi, coby součást propagandy.

Neschopen cokoliv veřejně zkritizovat, zvolil utajenou konverzaci pomocí mrkání. Během přenosu Jeremiah vymrkal slovo “torture”.[3]



Obrázek 2.3: Časová osa vývoje steganografie. [29]

[2] pomůcka nutná pro rozšifrování zprávy, zpravidla kus papíru,nebo kartónu s děrami
[3] z ang. - mučení

2.2.7 Různé techniky steganografie napříč historií

Klasický model pro skrytou komunikaci poprvé navrhl Simmons - `Prisoner's problem` [19].

Prisoner's problem

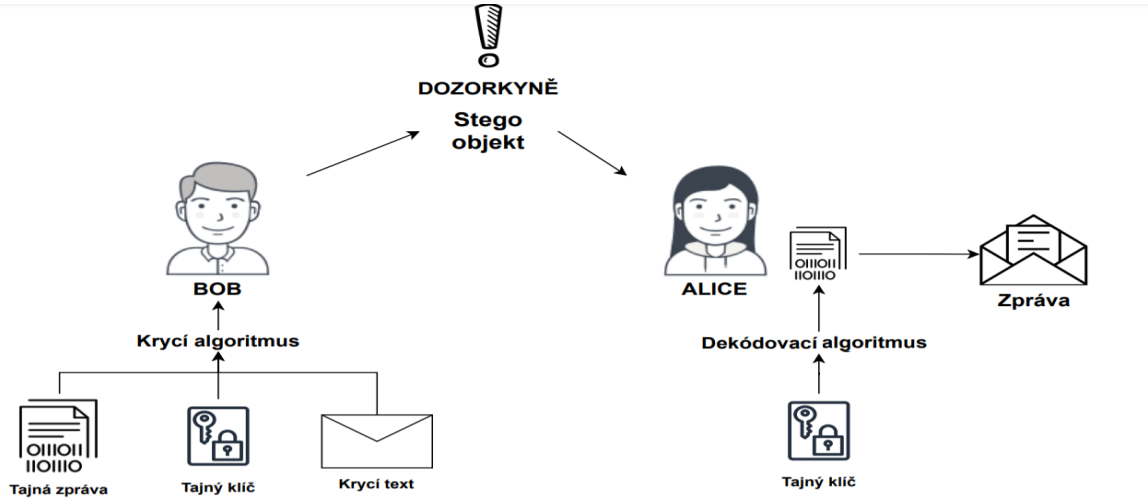
Tento referát byl vytvořen roku 1983 G.J. Simmonsem pro lepší pochopení problematiky tajné komunikace [30].

Dva vězni Bob a Alice jsou uvězněni odděleně a jediná možnost jejich komunikace je přes dozorkyni. Pokud pojme dozorkyně podezření na jakoukoliv nepovolenou činnost, umístí je na samotku [19].

Jediný způsob tajné komunikace je nepozorovatelná změna textu nebo obrázku. Bob proto nakreslí obrázek s modrou krávou a zelenou pastvinou, jenž obsahuje skrytou zprávu. Dozorkyně bez podezření předá obrázek, aniž by si byla vědoma jakékoliv zašifrované zprávy.

Tento způsob tajné komunikace bohužel není stoprocentně spolehlivý a má své slabiny. Pokud by dozorkyně upravila obrázek, mohla by pozměnit, nebo zničit zašifrovanou zprávu. Dalším typem útoku na zašifrovanou komunikaci je tzv. `malicious attack`, v tomto případě by dozorkyně vytvořila vlastní zprávu a předstírala, že je od Boba. Dozorkyně by také mohla zprávu od Boba zapomenout a Alici by jí řekla vlastními slovy, opět by byla tajná zpráva poškozena - tzv. `pasivní útok`.

Tento model popisuje obecný princip steganografie, vždy bude dvě a více tajně komunikujících stran a naslouchající, snažící se utajenou komunikaci zachytit. Proto je potřeba při využívání steganografie pro tajnou komunikaci brát zřetel na potenciální útoky.



Obrázek 2.4: Znázornění obecného principu šifrování zpráv pomocí steganografie. Stego objekt značí médium obsahující skrytou zprávu.

Mikrotečky

Tento nacistický vynález se běžně používal během 2. světové války. Spočívá v pořízení fotografie textu, která je následně zmenšena přibližně na velikost poštovní známky. Následně je reverzním mikroskopem zmenšena na velikost jednoho milimetru (zmenšení až 1:200) [30]. Pomocí jehly se poté obrázek nanese na negativ.

Vernamova šifra

Spolehlivým ukrytím tajné zprávy je tzv. Vernamova šifra (v angličtině One-Time Pads). Každý znak zprávy je posunut o náhodný počet míst abecedy.

Tento systém kódování je považován za nerozluštitelný, za předpokladu, že každý klíč je zcela náhodný a je použit maximálně jedenkrát.

Nulová šifra

Princip nulové šifry spočívá v ukrytí tajné zprávy do většího množství cover textu na základě jednoduchého vzoru. Nevýhodou této metody je snadná rozluštitelnost, navíc text nemusí vždy znít dobře.

```
Fishing freshwater bends and saltwater coasts rewards anyone
feeling stressed. Resourceful anglers usually find masterful
leapers fun and admit swordfish rank and overwhelming any day [12].
```

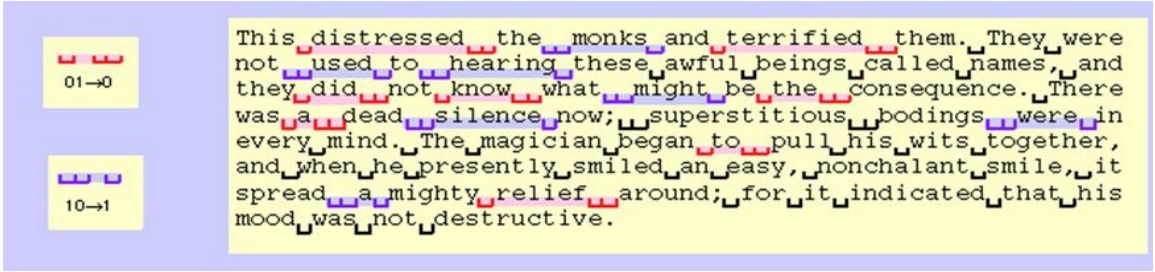
V textu je pomocí nulové šifry schovaná tajná zpráva “Send lawyers guns and money”. V tomto případě je zpráva ukrytá na pozici třetího písmena každého slova, avšak vzor může být libovolný. Která písmena z textu vybrat si komunikující strany musí domluvit předem.

Tato metoda opět našla své uplatnění během 2. světové války [30].

Mezery a Odsazení

Tato forma kódování chytře a nenápadně upravuje původní text ke skrytí tajných dat. K zakódování zprávy stačí upravit odsazení řádků, nebo přidat mezeru mezi slova. Tato slova nacházející se mimo jejich běžnou pozici indikují tajnou zprávu.

Nadbytečné mezery a odřádkování reprezentují v binární soustavě[4] 0 a 1.



Obrázek 2.5: Vložení tajné zprávy pomocí odsazení slov (převzato z [8]).

Rozprostřené spektrum

Metoda rozprostřeného spektra vezme malý zvukový signál a smíchá ho s širším přenosovým signálem[5]. Tato metoda tajné komunikace je velmi robustní, jelikož používá velké množství krycího signálu, který obklopuje tajnou zprávu.

[4] Číselná soustava o základu 2. Využívá pouze číslice 0 a 1.

[5] Šířka pásma zprávy výrazně překračuje koherenční šířku pásma kanálu

Techniky zkreslení

Ukládají informace podle signálu a měří odchylku od původního textu v jednom kroku dekódování [19].

Neviditelný inkoust

Princip této metody spočívá v použití "neviditelných" tekutin, které jsou viditelné pouze po zahřátí, pod UV světlem, nebo po použití speciálních chemikálií. V průběhu historie se pro tyto účely využívalo velké množství tekutin, například mléko, ocet, citrónová šťáva i lidská moč.

Tento způsob tajné komunikace byl často využíván pro válečné i politické účely napříč celou historií. Metodu používal například i George Washington v 2. polovině 18. století pro komunikaci se vzdálenými špehy [30].

Používání žargonu

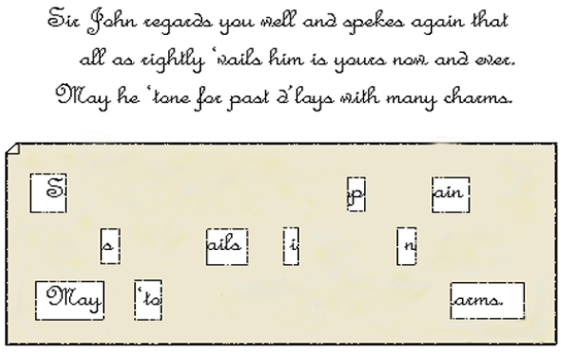
Tato metoda se užívala během 2. světové války pro snadné maskování zpráv. Metoda spočívá v nahrazování celých slov.

Během 2. světové války se k FBI[6] dostal dopis "Polámané anglické panenky budou v nemocnici pro panenky pár měsíců, dokud nebudou opraveny. V nemocnici pro panenky pracují ve dne i v noci". FBI se podařilo zjistit, že panenky reprezentují lodě. Každý druh panny představoval určitý typ lodě. Nemocnice pro panenky značila dok [30].

Cardanova mřížka

Tento způsob poslání tajné zprávy se poprvé objevil již ve starověku, kdy se používala deska z pevného materiálu.

Tato metoda byla převzata vynálezcem jménem Girolamo Cardano, po kterém je pojmenována. Každý z příjemců má kus kartónu s děrami uvnitř, které po překrytí nevinně vypadající zprávy odhalí tajnou zprávu.



Obrázek 2.6: Ukázka odhalení tajné zprávy po přiložení Cardanovy mřížky (převzato z [28]).

Tato metoda se hojně využívala při politické korespondenci během 16. a 17. století [30].

[6] Americký Federální úřad pro vyšetřování. Z ang. Federal Bureau of Investigation

Steganografický souborový systém

Systém si vezme velké místo na disku, zašifruje a schová data tak, že bez hesla nelze dokázat, že existují.

Pro správnou funkčnost musí být hesla v hierarchii. Pokud máme 3 soubory, uživatel s přístupem k třetímu souboru musí mít také přístup k souborům 1 a 2 [27].

Tento systém nám umožňuje schovávat dokumenty v jiných zdánlivě náhodných souborech. Navíc nám zajišťuje dvouúrovňovou ochranu. Zašifrované soubory jsou viditelné a stego soubory jsou "neviditelné" [27].

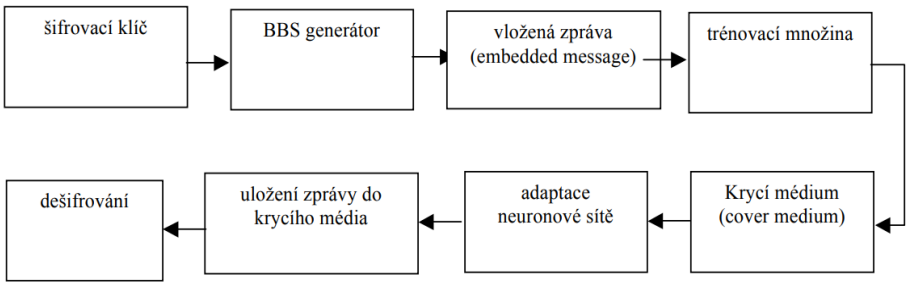
2.3 Současný stav steganografie

V současnosti by se mohlo zdát, že je steganografie přežitek. V dnešní digitální době, plné výkonných počítačů a zařízení by se kryptografie mohla jevit jako lepší a bezpečnější řešení pro přenos zpráv. Pokud ale pro přenos tajné zprávy zvolíme kryptografii, všichni okamžitě vědí, že se jedná o tajnou komunikaci a ta se rázem stane cílem jejich útoku.

To je hlavní a největší výhoda steganografie, umožnit odesílateli nepozorovaně doručit tajnou zprávu. Proto je i dnes steganografie hojně využívána. Nechvalně známým příkladem je používání obrazové steganografie teroristickou organizací Al-Kaidá, která posílá tajné informace svým zaoceánským členům pomocí pornografických obrázků [11].

2.3.1 Steganografie s využitím neuronových sítí

Odhalení tajné zprávy ukrytou steganografickou metodou není snadná záležitost, proto je zpráva dekódována neúspěšně ve více než 20 % případů [13]. Z tohoto důvodu je vhodné využít některého z modelů umělé inteligence. V kryptografické oblasti je stále populárnější využití neuronových sítí. Neuronové sítě se dají využít jak k šifrování souborů steganografickými metodami, tak k jejich dešifrování.



Obrázek 2.7: Steganografický systém využívající neuronových sítí (převzato z [10]).

Neuronové jednotky jsou různými modely spojovány do sítí. Každá síť je navržená pro jiné úlohy. Tyto sítě využívají několika různých algoritmů, například Backpropagation, Pruningův algoritmus, gradientní metody a další. [9]

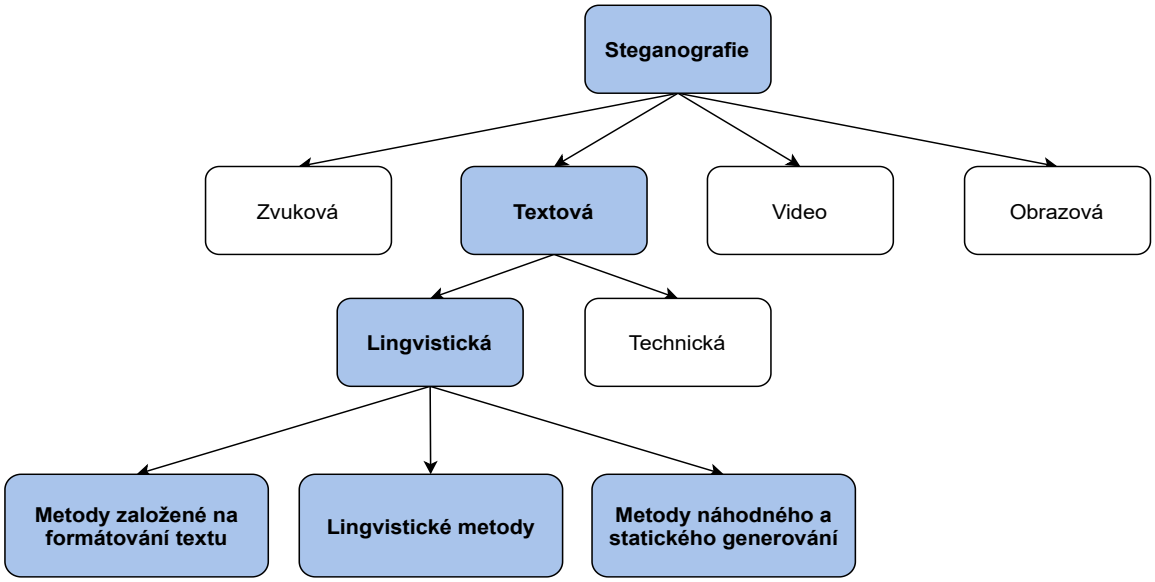
Nejčastěji jsou neuronové sítě využity pro obrazovou a zvukovou steganografii, kde jsou schopné odhalit ukrytou správu téměř ve 100 % případů [9, 13]. Metodám využívajících neuronových sítí se výrazně zlepšila účinnost z hlediska zachování utajení a kvality zakódované zprávy [5]. V nedávné době se navíc zjistilo, že neuronové sítě lze použít ke kódování více zvuků do jednoho krycího souboru.

2.4 Digitální textová steganografie

Textová steganografie využívá textové soubory jako krytí pro přenos tajných dat. Textová steganografie je považována za nejtěžší druh steganografie. Postrádá totiž veliké množství redundantních dat ve srovnání s digitálními médii jako je obraz, zvuk, nebo video [19].

Blíže se budu zabývat textovou steganografií lingvistickou, tj. taková, která k ukrytí zprávy nepotřebuje žádné speciální nástroje, nebo zařízení [12].

Digitální textová steganografie se dále dělí na 3 části: metody založené na formátování textu, lingvistické metody a metody náhodného a statického generování [21].



Obrázek 2.8: Rozdělení steganografie podle prostředí, ve kterém jsou ukrytá data a podle způsobu jejich zašifrování. Zvýrazněná je oblast, kterou se budu podrobně zabývat.

2.4.1 Metody založené na formátování textu

Metody založené na formátování textu libovolně upravují grafickou podobu textu za účelem vložení tajné zprávy. Všechny metody založené na formátování vkládají do cover zprávu pomocí binárních dat[7]. [12]

Existuje spousta druhů formátování textu pro účely ukrytí zprávy nepozorovaně. Mezi hlavní patří například přidání mezer, odsazení mezi řádky, změna fontu, či velikosti písma. Opět se snažíme, aby byly změny co nejméně nápadné běžnému čtenáři.

Open-space method

Tato metoda využívá přidávání bílých znaků[8] do cover textu.

Open-space metoda má několik způsobů provedení. Prvním je vložení bílých znaků mezi věty, tento způsob není efektivní. Aby tajná zpráva nebyla odhalena, můžeme za větu přidat velmi malý počet bílých znaků navíc. Pro ukrytí informace proto potřebujeme velké množství krycího textu. [12]

[7] Hodnota složená pouze z nul a jedniček. Binární hodnotou lze vyjádřit cokoliv.

[8] Netištěné znaky, tj. mezery

Dalším způsobem jsou mezislovní mezery. Metoda mění počet mezer mezi slovy za účelem ukrytí binární zprávy. Jedna mezera představuje “0”, dvě mezery představují “1”. [12]

Nejefektivnější variantou Open-Space metody je přidávání mezer na konec řádků. Tato metoda je hůře zpozorovatelná a zároveň nám umožňuje ukrytí většího množství tajných bitů.

Zarovnání textu

Zarovnání cover textu doprava je také používáno k zakódování. Princip opět spočívá v počítání mezer mezi slovy (viz 2.4.1). Tato metoda je velice efektivní a těžká na dekódování. Má ale jednu velikou nevýhodu. Jak poznat nevinnou mezeru od té zakódované, která má značit nulový bit? [23]

Řešením je tzv. `kódování Manchester`, které tuto nevýhodu eliminuje. Bity “0” a “1” jdoucí po sobě značí “1”. Bity “1” a “0” reprezentují “1”. [23]

Baconova šifra

Tato metoda vytvořena Sirem Francisem Baconem, využívá 5 bitů (BSCII[9]) pro kódování 24 písmen abecedy (bez “j” a “u”) od 00000 do 10111. Číslice “0” nahrazuje písmenem “A” , číslice “1” nahrazuje písmenem “B” (takže od AAAAA po BABBB). [7]

Ukázka použití Baconovy šifry.

Princip ukrytí zprávy je následující. Máme 2 fonty písma, první značí písmeno “A” , druhý font písmeno “B”. V reálné situaci by si tyto fonty měly být co nejpodobnější, pro lepší demonstraci metody, jsem použil výrazně odlišné.
Z textu lze získat schéma: **BAABA AAAA ABAAA ABBAA AABAA**
Pomocí následující tabulky převedeme schéma na tajnou zprávu: **tajne**

a	AAAAA	e	AABAA	i,j	ABAAA	n	ABBAA	r	BAAAA	w	BABAA
b	AAAAB	f	AABAB	k	ABAAB	o	ABBAB	s	BAAAB	x	BABAB
c	AAABA	g	AABBA	l	ABABA	p	ABBBA	t	BAABA	y	BABBA
d	AAABB	h	AABBB	m	ABABB	q	ABBBB	u,v	BAABB	z	BABBB

Tabulka 2.2: Tabulka pro Baconovo šifrování

Kódování vlastností

Tato metoda se využívá zejména v MS Word souborech. Metoda ukrývá tajnou zprávu pomocí úprav textu, které jsou pro lidské oko neviditelné. Patří zde například změna měřítka znaků, ohraničení odstavců, ohraničení symbolů a různé způsoby vkládání dat. [4]

Formátování CSS

Formátování textu neznamená pouze formátování .docx souborů. Formátovat můžeme také XML, Html, CSS a další. Tato metoda spočívá v ukrytí tajné zprávy pomocí změny CSS souboru.

[9] Baconův standard

Metoda nejprve zašifruje tajnou zprávu pomocí veřejného klíče RSA a následně šifruje text. Metoda spočívá v přidávání bílých znaků na konec řádku, díky čemuž je metoda špatně postřehnutelná. Mezera bezprostředně za středníkem reprezentuje "0" bit, tabulátor reprezentuje "1" bit. [2]

Formátování XML

Odlišnou formu steganografie představují steganografické metody formátující XML. Celý XML soubor zde představuje tajnou zprávu a metody jej mění takovým způsobem, aby zpráva nebyla patrná. Lze toho dosáhnout metodou `míchání značek (XML tagů)`. Prohozením prvního tagu a posledního, druhého a předposledního atd. Poté co projdeme první úroveň v hierarchii XML tagů aplikujeme stejný postup na další úrovně. [22]

Další variantou je `specifikované míchání značek`. Princip zůstává stejný jako u předchozí metody, s tím rozdílem, že pořadí značek je určené hodnotou atributu.

2.4.2 Lingvistické metody

Princip metod lingvistické steganografie spočívá ve změně struktury textu. Může se jednat o narušení gramaticky správné syntaxe[10], nebo sémantickou záměnu slov. [12]

Sémantické metody

Princip ukrytí informace je podobný jako u všech ostatních metod. Cílem je do textu nepozorovaně ukrýt sekvenci bitů, které představují skrývanou informaci. Sémantické metody k tomu používají záměny slov.

Jednou z nejznámějších je sémantická metoda, kterou vynalezl Mohammad Hassan Shirali-Shahreza. Tato metoda využívá záměnu výrazů pomocí synonym[11]. Při použití této metody vyžadujeme předem danou tabulku, která bude specifikovat, které slovo z dvojice představuje bit "0" a "1". Sémantická metoda je velmi spolehlivá a text s ukrytou zprávou v žádném případě nevypadá podezřele. [21]

Slovo	Synonymum
Hard	Difficult
Movie	Film
Unhappy	Sad
Cookie	Biscuit

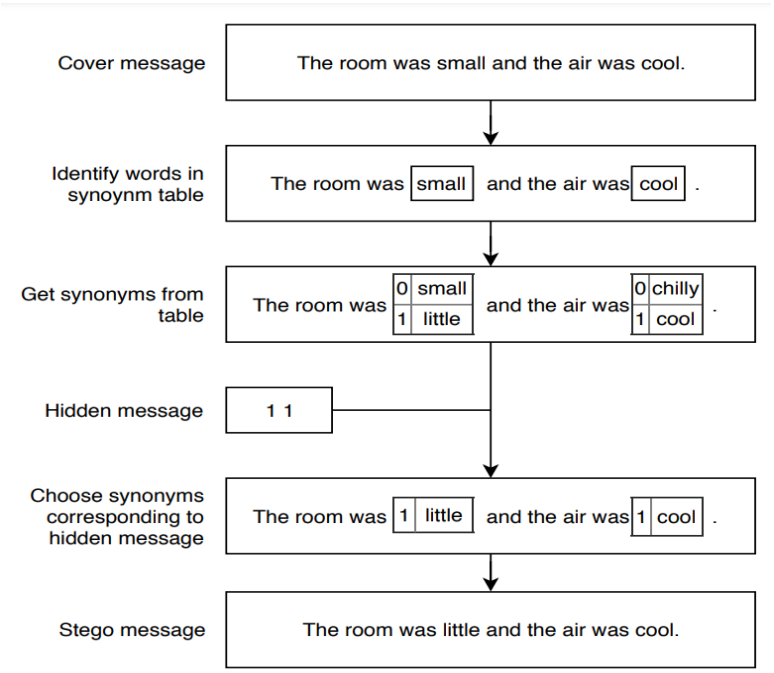
Tabulka 2.3: Příklady synonym

Další podobná metoda zaměňuje výrazy z britské a americké angličtiny [21]. Tuto metodu na rozdíl od předchozí lze využít pouze v anglicky psaných textech. Rodilý mluvčí, nebo člověk zdatný v anglickém jazyce, může velmi rychle nabýt dojmu, že text není v pořádku.

[10] Soubor pravidel definující správné kombinace slov ve větné stavbě.
[11] Slovo se stejným, nebo podobným významem.

Britské výrazy	Americké výrazy
Flavour	Flavor
Neighbour	Neighbor
Humour	Humor
Dialogue	Dialog
Colour	Color
Cheque	Check

Tabulka 2.4: Velmi podobné britské a americké výrazy vhodné k záměně slov



Obrázek 2.9: Princip ukrytí tajné zprávy pomocí metody synonym (převzato z [6]).

Existuje nespočetně mnoho sémantických metod založených na podobném principu. Slova můžeme zaměnit za zkratky, smajlíky, protiklady a další.

SMS forma

Tyto metody využívají kombinaci zkrácených slov používaných v SMS. Pro implementaci této metody je potřeba vytvořit slovník, který obsahuje slova a k nim odpovídající SMS zkratky. Zkrácený tvar značí bit “1” , zatímco plný tvar slova značí bit “0”. [2]

Syntaktické metody

Ke skrytí tajné zprávy do cover textu lze například využít manipulace s interpunkcí [2]. Velkou nevýhodu této metody je, že metoda musí identifikovat vhodná místa pro vložení těchto znaků.

```
červená, zelená, a modrá
červená, zelená a modrá
```

Nadbytečné použití interpunkce je velmi nápadné, a proto tato metoda není vhodná k ukrytí zprávy.

Jiné syntaktické metody se zaměřují na změnu dikce[12] a pořadí slov ve větné stavbě. Tento způsob je mnohem bezpečnější, ale obtížněji proveditelný.

Tvorba vět

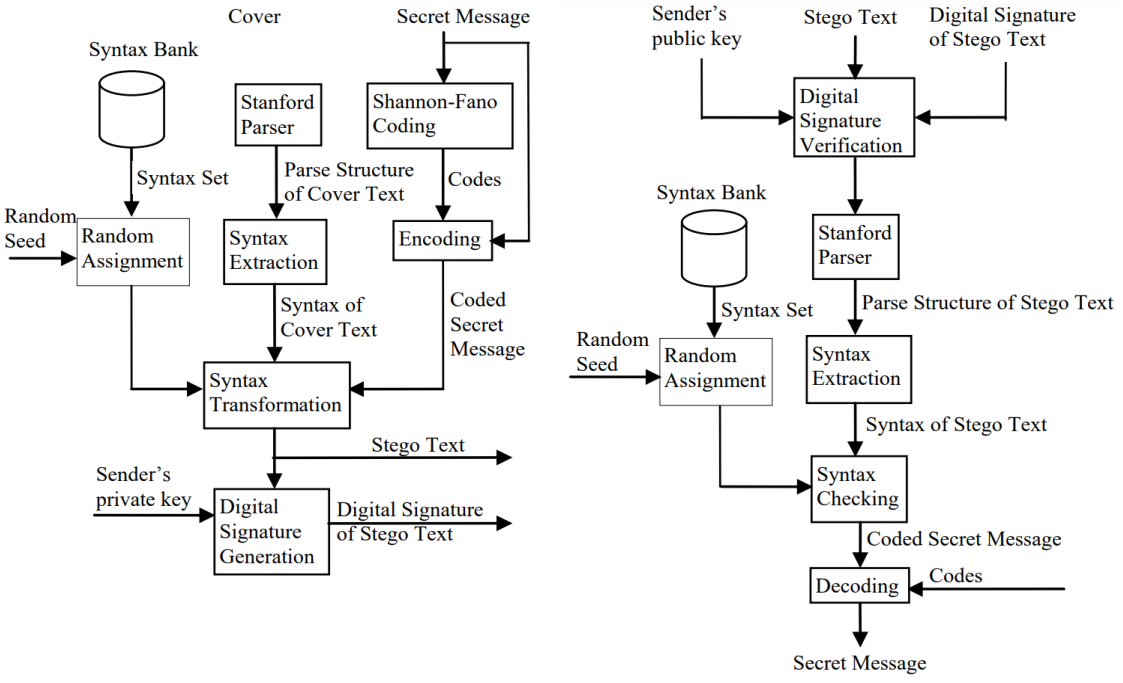
Mnohé z lingvistických metod skrývají tajné informace pomocí tvorby vět. Větu lze přetvořit bez změny jejího významu. Nejsnadnější je v anglickém jazyce změna trpného a činného rodu (we have received the goods → the goods have been received).

Tvorba věty se skládá z několik fází. Nejprve je text komprimován pomocí `Shannon-Fano` algoritmu, který vytváří prefixový kód na základě pravděpodobnosti [26]. Poté je věta transformována do jedné z forem syntaxe.

Cover text je parsován pomocí `Standfordského analyzátoru` [26]. Tento analyzátor využívá znalosti jazyka získané z ručně analyzovaných vět k vytvoření co nejpravděpodobnějších nových vět. Výstup analyzátoru je použit jako vstup `syntaktické transformační fáze`. [25]

Syntaktická transformační fáze využívá takzvanou `syntaktickou banku`. Tato banka se skládá ze syntaktických sad, které představují všechny dostupné syntaktické formy věty. Gramatická struktura vytvořená Standfordským analyzátorem je převedena do syntaktické formy. Poté je komprimovaná tajná zpráva aplikována na větu. [26]

Odesílatel a příjemce ještě před začátkem mezi sebou sdíleli klíč potřebný k vytvoření stejné náhodné sekvence správného pořadí, které odpovídá syntaktickým pravidlům. [25]



Obrázek 2.10: Znázornění ukrytí a dešifrování zprávy pomocí změny syntaktické stavby vět (převzato z [25]).

[12] Způsob vyjadřování a volba výrazových prostředků

2.4.3 Metody náhodného a statistického generování

Aby se steganografové vyhnuli práci se známým textem, často si generují vlastní cover text pro ukrytí šifrované zprávy.

Výhoda těchto metod je, že třetí strana nezná cover text, avšak velkou nevýhodou je, že tento vygenerovaný text nemusí vždy působit zcela legitimně.

Jeden z nejlepších příkladů náhodného generování steganografického textu je aplikace `spammimic`[13], tento nástroj využívá "spamovou" gramatiku a Waynerův algoritmus pro bezkontextovou gramatiku. Zpráva je tvořena ze spamových frází. Jedná se o přesvědčivou formu generování textu, jelikož spousta nesmyslného spamu je volně dostupná všude na internetu. A skutečný spam je většinou napsaný tak hloupě, že je téměř nemožné rozeznat náhodně vygenerovaný spam od toho skutečného. V nástroji `spammimic` se vygenerované fráze liší na základě zprávy, kterou chceme zašifrovat. [20]

Mimické funkce

Mimické metody využívají bezkontextovou gramatiku[14]. Generovaný text se snaží napodobovat text reálný, použitím statistiky výskytu slov a písmen, délky slov a četnosti jednotlivých znaků pro vygenerování velmi realistických vět. Při každém kroku vybere metoda jednu možnost ze získaných dat.

Kódování:
`Start` → `podmět, přísudek`
`podmět` → `Petr | Pavel`
`přísudek` → `poslal | odeslal`
`předmět` → `email | sms`

Tato tabulka nám umožňuje vytvořit 8 různých vět, přičemž každá věta v sobě ukrývá jinou posloupnost ukrytých bitů. (Petr poslal sms: **011**, Pavel odeslal email: **110** atd.). Je patrné, že počet vygenerovaných možností je dán vztahem 2^N, kde N je počet pravidel v dané tabulce. [20]

2.5 Vybrané vlastnosti steganografických metod

Porovnání jednotlivých metod je vytvořeno na základě existujících publikací [1, 2], obsahujících šifrování textu pomocí těchto metod. Přesnému vyhodnocení a porovnání těchto steganografických metod se budu podrobně věnovat v části `Postřehnutelnost` (viz 5).

2.5.1 Postřehnutelnost

Postřehnutelnost je v oblasti steganografie nejdůležitější vlastností. Celý princip steganografie staví právě na nepostřehnutelnosti. To znamená, že všechny metody usilují o co nejmenší postřehnutelnost.

Čím je zašifrovaný text více postřehnutelný, tím více působí upraveně oproti originálnímu textu, a tím více budí ve čtenáři podezření. Naopak text upravený metodou, která je špatně postřehnutelná, by měl čtenář vnímat jako naprosto běžný, neupravený text.

[13] https://www.spammimic.com/index.cgi
[14] Množina pravidel. Pravidla tvoří různé řetězce (věty)

Postřehnutelnost	
Open-space method	Velmi nízká
Baconova šifra	Vysoká
Metoda synonym	Nízká
Nulová šifra	Vysoká
Mimické funkce	Nízká
Kódování vlastností	Velmi nízká

Tabulka 2.5: Postřehnutelnost vybraných metod.

Existující publikace [2] z vybraných metod jako nejméně postřehnutelnou označily Open-space metodu. Naopak jako téměř nepoužitelnou označily Baconovu šifru. Která metoda je skutečně nejméně postřehnutelná se dozvíte v sekci "Postřehnutelnost" (viz 5.2.6).

2.5.2 Robustnost

Robustnost, neboli náchylnost na změnu při manipulaci s textem je další faktor při hodnocení steganografických metod. Ke změně struktury textu, která by poškodila ukrytou zprávu, může dojít i neúmyslně, proto je robustnost u steganografických metod velmi důležitá vlastnost.

Robustnost	
Open-space method	Velmi nízká
Baconova šifra	Nízká
Metoda synonym	Velmi vysoká
Nulová šifra	Nízká
Mimické funkce	Vysoká
Kódování vlastností	Nízká

Tabulka 2.6: Vyšší robustnost implikuje lepší využití steganografické metody.

Robustnost je testována změnou výstupního textu, který již obsahuje skrytou zprávu. Text lze modifikovat přidáním/odebíráním slov, změnou fontu, nebo velikosti písmen. Při dekódování nás zajímá, jaký vliv měly změny na podobu ukryté zprávy.

Pokud je steganografická metoda vysoce robustní, modifikace výstupního textu by neměly mít na ukrytou zprávu žádný vliv.

O tuto steganografickou vlastnost usilujeme, pokud víme, že se bude se zašifrovaným textem v budoucnu manipulovat.

2.5.3 Kapacita

Kapacita metod nám říká, kolik lze maximálně uložit bitů do daného souboru, za použití vybrané metody.

Kapacita = počet ukrytých bajtů/velikost cover textu[B]

Kapacita	
Open-space method	Velmi nízká
Baconova šifra	Velmi vysoká
Metoda synonym	Velmi nízká
Nulová šifra	Nízká
Mimické funkce	Vysoká
Kódování vlastností	Vysoká

Tabulka 2.7: Tabulka nám říká, která z metod dokáže ukrýt největší množství bitů pro stejně velký cover text.

Nejnižší kapacitu z vybraných metod má metoda synonym, ta pro ukrytí tajné zprávy potřebuje velké množství cover textu a rozsáhlé slovníky pro nahrazování slov.

Nízkou kapacitu má také Open-space metoda. Tato metoda nemůže ukrývat větší množství bitů, neboť větší množství mezer mezi slovy by bylo opravdu velmi nápadné. Avšak existuje modifikace této metody, která přidává bílé znaky na konec řádku. Tento způsob je, co se týče kapacity metody, mnohem lepší (viz 2.4.1).

Kapitola 3

Rozbor řešení a jeho návrh

3.1 Formát souboru a typ skrývané informace

Pro ukrytí zprávy do textového souboru lze využít mnoho textových formátů, například `.txt` `.docx` `.doc` `.adoc` a další. Já se rozhodl pracovat pouze s prvními 2 výše zmíněnými formáty. Výstupní soubor bude vždy ve formátu `.docx`, který mi umožní použití všech možných steganografických metod (.txt například neumožňuje změnu fontu jednotlivých slov).

Každá metoda má navíc vlastní znakovou sadu, kterou je ukrytá informace kódována. Nejčastěji využívaná sada je ASCII. V současnosti se častěji používá rozšířené ASCII, které kóduje znaky do 8 bitů (z původních 7). 8-bitové ASCII dokáže kromě písmen a číslic, zobrazit i velké množství speciálních znaků. Tuto sadu jsem se rozhodl použít u všech svých metod, kromě Baconovy šifry, jenž má své vlasní kódování.

3.2 Realizační prostředky

3.2.1 Implementační jazyk

Jako implementační jazyk jsem zvolil multiplatformní skriptovací jazyk `Python`, verze `3.8.1`. Jedná se o objektově orientovaný jazyk s jednoduchou syntaxí, která zaručuje snadno čitelný kód. Ekosystém jazyka Python je velmi bohatý, a pro tuto bakalářskou práci nabízí mnoho užitečných knihoven, jako je například `Matplotlib`[1] pro vykreslování grafů (viz 5), `docx, xml.etree.ElementTree` a další.

Knihovny jsou často nástavbami implementovanými v jazyce C/C++, proto tento jazyk zaručuje i vysokou rychlost programu.

Dalším faktorem pro výběr tohoto jazyka je snadná manipulace s textem, která představuje hlavní náplň této práce.

3.3 Aplikace

3.3.1 Průzkum existujících aplikací

Jako součást řešení jsem se rozhodl vytvořit aplikaci, pro lepší demonstraci jednotlivých steganografických metod. Při tvorbě návrhů své steganografické aplikace jsem vycházel z již

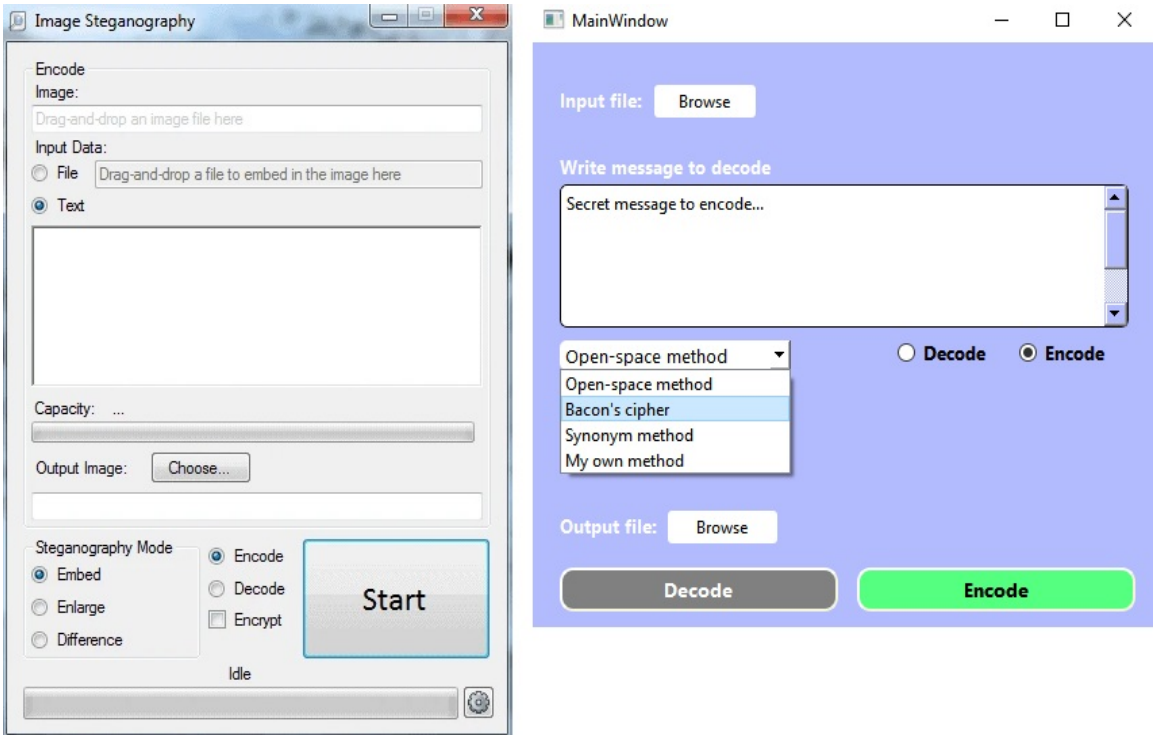
[1] https://matplotlib.org/

existujících aplikací, jako jsou například `Steghide`[2], `Hide'N'Send`[3], nebo `Xiao Steganography`[4].

3.3.2 Návrh GUI

Pro tvorbu uživatelského rozhraní jsem si vybral knihovnu `Tkinter`[5]. Jedná se o jeden z neoblíbenějších nástrojů pro tvorbu GUI, který je navíc vhodný i pro úplné začátečníky.

Aplikace umožní uživateli vybrat textový soubor, zvolit si jednu z implementovaných metod a zašifrovat do vybraného souboru zprávu, nebo ji naopak ze souboru dešifrovat.



Obrázek 3.1: Nalevo je aplikace pro obrazovou steganografii, kterou jsem se inspiroval [14], napravo je prvotní návrh mojí aplikace.

3.4 Návrh řešení

3.4.1 Vybrané metody textové steganografie

V rámci praktické části této bakalářské práce jsem se rozhodl implementovat následující metody z oblasti steganografie. Dvě z nich jsou založené na formátování textu **Open-space method** (varianta s mezerami mezi slovy, viz 2.4.1)
a **Baconova šifra** (viz 2.4.1). Z oblasti lingvistických metod jsem si zvolil **Metodu synonym** (viz 2.4.2). Na závěr se pokusím implementovat vlastní metody vycházející z těchto metod.

[2] http://steghide.sourceforge.net/
[3] https://www.softpedia.com/get/Security/Encrypting/Hide-N-Send.shtml
[4] https://xiao-steganography.en.softonic.com/
[5] https://docs.python.org/3/library/tkinter.html

Všechny tyto metody pozměňují cover text pro ukrytí bitů obsahujících tajnou zprávu. Tyto metody jsem záměrně vybral podle odlišného stylu kódování a dekódování tajné zprávy.

Dalším kritériem pro výběr těchto metod byly jejich přednosti. Vybral jsem tedy metody takové, z nichž každá vyniká v jiném aspektu steganografie. Některá metoda dokáže ukrýt mnohem větší množství bitů než jiné, jiná je odolná vůči všem možným změnám textového souboru. Použití některých metod si všimneme okamžitě, jiné nemáme šanci postřehnout. Porovnání těchto metod se věnuje jiná část práce (viz 5.4).

3.4.2 Vlastní metoda

Součástí této práce je i navržení vlastní steganografické metody. Na poli steganografie je velmi obtížné přijít s nápadem na novou metodu. Existuje enormní množství nejrůznějších metod a všechny dobré nápady byly již vyčerpány. Proto svou vlastní metodu vytvořím z již existujících metod. Skloubením jejich principů vytvořím novou metodu, která bude kombinací jejich nejsilnějších stránek.

Baconova šifra + Open-space metoda

Prvním příkladem je šifrování Baconovou šifrou, u kterého jsou pro nás relevantní pouze řádky s mírně pozměněnou hodnotou odřádkování (Open-space metoda). Vzniklá metoda by tedy kombinovala přednosti obou metod, špatnou postřehnutelnost a dobrou kapacitu pro ukrytí zprávy.

Baconova šifra + metoda synonym

Mým dalším návrhem je kombinace Metody synonym a Baconovy šifry. Metoda synonym by měla mít ze všech metod nejnižší kapacitu, což kompenzuje vysokou robustností. Proto navrhuji použít způsob šifrování dat, které využívá Baconova šifra k zvýšení kapacity metody synonym.

Huffmanovo kódování

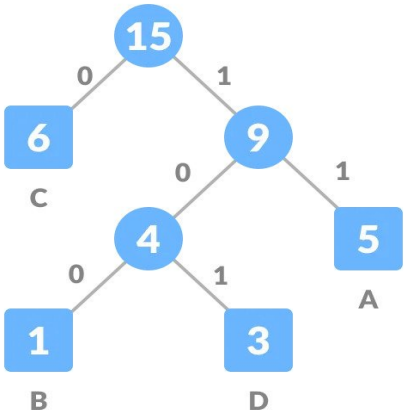
Zvýšit kapacitu metod by šlo i dalšími způsoby. Velmi oblíbené je například Huffmanovo kódování, které se běžně používá pro bezeztrátovou kompresi dat. Princip šifrování s tímto typem kódování by probíhal následovně. V tajné zprávě, kterou si přejeme zašifrovat, si zjistíme nejčastější výskyty znaků, nebo posloupností znaků. Následně vytvoříme Huffmanův strom, který častěji používaným znaků, přiřadí kratší bitový vzor.

Znak	A	B	C	D
Četnost	5	1	6	3
Huffmanův kód	**11**	**100**	**0**	**101**

Tabulka 3.1: Znaky (caadddccacacac) byly za pomocí Huffmanova kódování převedeny na kratší sekvence bitů.

Výsledný kód `1000111110110110100110110110` vytvořený na základě předchozí tabulky se nazývá prefixový. To znamená, že žádné kódové slovo není prefixem slova jiného.

Časová složitost zakódování znaku na základě jeho četnosti v tajné zprávě je `O(nlog n)` [16].



Obrázek 3.2: Huffmanův strom pro vstupní řetězec bcaadddccacacac (převzato z [16]).

3.4.3 Automatizované testy

Existující publikace pro zhodnocení vlastností steganografických metod provádí testování všech klíčových vlastností nad vybraným textovým datasetem[6]. Žádné z těchto benchmarkových dat/datasetů jsem při hledání nenašel. Důvodem může být, že existuje velké množství steganografických metod, z nichž každá má jiný způsob ukrytí informace, bylo by tedy potřeba mnoho různých testovacích sad. Dalším důvodem je, že je velmi snadné vytvořit si vlastní cover text.

Jelikož jsem nenašel žádné konkrétní testovací soubory, vytvořil jsem si vlastní testovací data, na míru svým metodám. Například nebudu testovat metodu synonym na jiných než anglických testech, takovéto testy by negativně zasáhly do výsledků této metody. V existujících publikacích často testovali steganografické metody pomocí spamu. Mé testy se skládají kromě spamových souborů také z textů písní, elektronických knih a nejrůznějších volně dostupných souborů.

Testy důkladně prověří základní steganografické vlastnosti implementovaných metod. V plánu mám testovat `rychlost metod, úspěšnost ukrytí, kapacitu a robustnost.` Na některé důležité vlastnosti, jako například `postřehnutelnost` nelze napsat testy. Vyhodnocení této vlastnosti bude na základě mého osobního dojmu a názoru dotázaných osob.

Rychlost metod

Testy na rychlost nám řeknou, která metoda dokáže nejrychleji zašifrovat všechny poskytnuté soubory. K vyhodnocení této vlastnosti využiji knihovny `timeit`.

```
1    #soucasny adresar + presun do slozky s cover soubory
2    path = os.getcwd() + '/cover_files/synonyms'
3    list_of_files = os.listdir(path)
4    #zacatek mereni
5    start = timeit.default_timer()
6    for file in list_of_files:
7        #sifrovani se vzolenymi vstupnimi argumenty
8        steganography.main(['-i', file, '-e', '-s', secret_message, '-r'])
```

[6] kolekce dat

```
9     #konec mereni
10    end = timeit.default_timer()
11    time = (end - start)
```

Úspěšnost ukrytí

Úspěšnost steganografické metody nám udává v procentech, do kolika souborů z celkového počtu se podařilo vložit celou tajnou zprávu.

$$\text{Úspěšnost} = \frac{\text{počet zašifrovaných souborů (obsahující celou zprávu)}}{\text{celkový počet cover souborů}} * 100\ [\%]$$

Kapacita metod

Kapacita nám udává množství dat, které je možno skrýt do vstupního textu. [4]

```
1     #kompletni text cover souboru
2     full_text = steganography.print_text(file)
3     if method is "synonyms":
4         words_available = synonyms.count_dictionary_words(full_text)/8
5     elif method is "bacon":
6         words_available = len(full_text.split())/5
7     elif method is "spaces":
8         words_available = len(full_text.split())/8
```

Výpis 3.1: Zjištění maximálního počtu slov, který lze ukrýt.

Pro výpočet kapacity je zásadní vědět, kolik znaků je maximálně metoda schopna ukrýt do daného souboru. Kapacitu následně spočítám jako poměr maximálního počtu ukrytých bajtů, ku velikosti souboru (viz 2.5.3).

3.4.4 Robustnost

Robustnost udává míru schopnosti metody zachovat původní ukrytou zprávu. V souboru zašifrovaným vysoce robustní metodou, zůstane skrytá zpráva zachována i po manipulaci s tímto souborem.

Tuto vlastnost budu testovat následovně. Vytvořím si několik formátovacích funkcí, z nichž každá bude jiným způsobem manipulovat s zašifrovaným textem. Měnit budou například velikost písma a jeho font, hodnotu odřádkování, budu přidávat odstavce a různě měnit strukturu souboru.

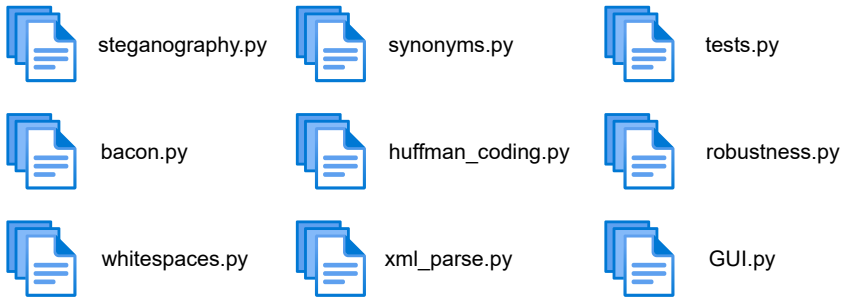
Zajímat mě bude, která funkce je odolná vůči jakým změnám. Tzn., po kterých formátovacích úpravách zůstane vložená tajná zpráva nepoškozená.

Kapitola 4

Implementační část práce

4.1 Implementované metody a struktura programu

Implementovány byly všechny metody zmíněné v sekci návrh řešení (viz 3.4). Implementace každé metody se nachází zvlášť v jednotlivých souborech.



Obrázek 4.1: Soubory obsahující implementované řešení.

Soubor `steganography.py` obsahuje zpracování vstupních argumentů, převod textového souboru .txt na formát .docx a další pomocné funkce potřebné pro řešení (převod binárních dat do čitelné podoby, výpis obsahu dokumentu a další).

Soubory `bacon.py, synonyms.py` a `whitespaces.py` obsahují implementaci jednotlivých metod. Každý z těchto souborů obsahuje funkci `encode`, která ukryje tajnou zprávu do vstupního cover textu a funkci `decode`, která tuto zprávu dokáže extrahovat.

Soubor `xml_parse.py` je stěžejní pro zachování původního stylu dokumentu. Zde dochází k parsování původního dokumentu a kopírování jeho formátovacích vlastností (některé jsou pozměněné pro účely šifrování) do nového souboru.

Automatizované testy jsou implementovány v souboru `tests.py`, jednotlivé testy jsou zde rozděleny do oddělených funkcí. Testování robustnosti probíhá v souboru `robustness.py`.

Uživatelské rozhraní pro snadnou demonstraci výsledků se nachází v souboru `GUI.py`.

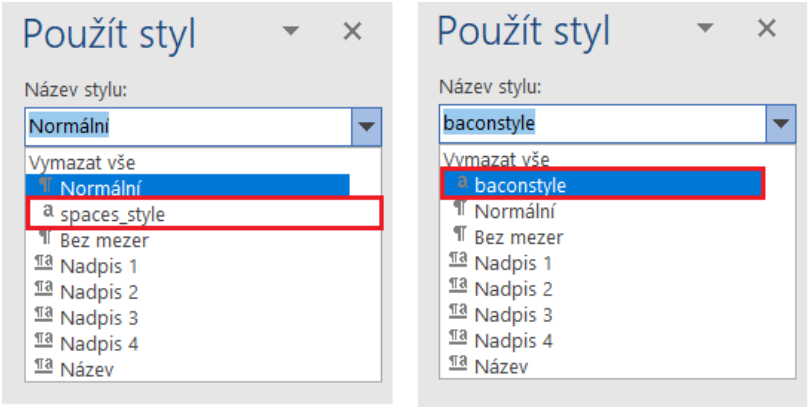
Více ke struktuře se dozvíte v příloze “Struktura datového média” (viz A).

4.2 Způsob implementace

Při implementaci zvolených metod jsem využil několika python knihoven, které mi značně usnadnily práci. Mezi ty nejdůležitější knihovny patří `docx`[1], `XML.etree.ElementTree`[2] a `numpy`[3].

4.2.1 Vytvoření vlastního stylu

Knihovna `docx` mi nejen umožní vytvořit dokument, ale navíc do něj i nahrát vlastní styl, což je nezbytné při použití metod, které formátují text.



Obrázek 4.2: Vlastní styly, které jsou použity pro vložení tajné zprávy.

Styly vytvářím ve funkcích `add_bacon_style` a `add_spaces_style`. Přiřadím k nim vlastnosti jako je velikost a font písma, a následně je styl přidán do dokumentu.

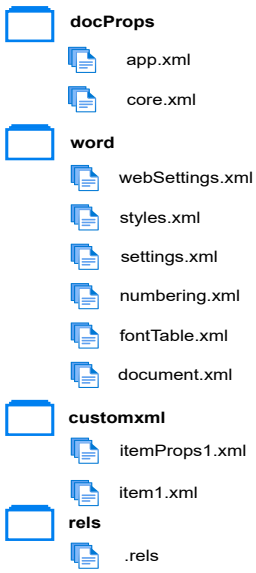
4.2.2 Zachování stylu původního dokumentu

Při snaze vložit utajenou zprávu do dokumentu jsem zjistil, že knihovna `docx` nedokáže přenést styl (formátování textu, font, velikost písma, odřádkování atd.) do nově vytvořeného dokumentu.

Tento problém mi pomohla vyřešit již zmíněná `XML` knihovna. Původní soubor `.docx` jsem rozbalil (podobná struktura jako `.zip` soubor) a dále pracoval pouze se souborem `word/document.xml`, který obsahuje hlavní textové prvky, včetně jejich stylu.

Ve funkci `split_document`, která se nachází v souboru `xml_parse.py` procházím původní XML soubor element po elementu a kopíruji jej do nového XML souboru. Text kopíruji včetně všech XML tagů, které nám udávají styl textu. Jednotlivé styly jsou potomky tagu `<w:rPr>`.

[1] https://python-docx.readthedocs.io/en/latest/
[2] https://docs.python.org/3/library/xml.etree.elementtree.html
[3] https://numpy.org/



Obrázek 4.3: Struktura dokumentu ve formátu .docx.

```
<w:r w:rsidRPr="00B52930">
  <w:rPr>
    <w:rFonts w:ascii="Agency FB" w:hAnsi="Agency FB" w:cs="Aharoni"/>
    <w:b/>
    <w:bCs/>
    <w:i/>
    <w:iCs/>
    <w:sz w:val="32"/>
    <w:szCs w:val="32"/>
  </w:rPr>
  <w:t>Lorem</w:t>
</w:r>
```

Obrázek 4.4: XML tagy

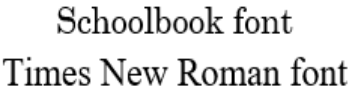
4.2.3 Implementace – Baconova šifra

Šifrování

Jak jsem již dříve vysvětlil (viz 2.4.1), princip Baconovy šifry spočívá v ukrývání bitů do textu, z nichž jsou tvořeny vzory, které značí příslušnou znakovou interpretaci.

Bity jsou touto šifrou nejčastěji ukryty využitím 2 stylů písma. Já se rozhodl k tomu přiřadit navíc změněnou velikost písma. Díky čemuž snížím šance na poškození tajné zprávy. To by mohlo nastat v případě, kdyby původní cover text již obsahoval font, který slouží pro ukrytí bitu "1".

Při tvorbě vlastního stylu jsem vycházel z faktu, že většina textů je psána fontem `Times New Roman`. Snažil jsem se tedy najít font, který je tomuto vzhledově nejbližší. Experimentováním s různými fonty písma jsem dokázal najít velmi podobný font, který je téměř k nerozeznání od zmíněného `Times New Roman`.



Obrázek 4.5: Styly použité pro Baconovu šifru.

Jako font písma, který bude sloužit k ukrytí jedničkových bitů jsem tedy zvolil `Century Schoolbook`, velikosti písma 10. Stejně jako u `Times New Roman` se jedná o takzvaný serifový font[4].

Na následujícím bloku kódu můžeme vidět vložení vlastního stylu do dokumentu.

```
font_charstyle = font_styles.add_style('baconstyle', WD_STYLE_TYPE.CHARACTER)
font_object = font_charstyle.font
font_object.size = Pt(10)
font_object.name = 'Century Schoolbook'
```

Poté, co jsem funkcí `add_bacon_style` vytvořil vlastní styl, jej potřebuji aplikovat na vhodná místa. Uživatel si na vstupu zvolí tajnou zprávu, kterou si přeje ukrýt. Zprávu si převedu do binární podoby a dále pracuji pouze s XML.

Podobu původního cover textu zkopíruji do nového XML souboru a zároveň se dívám na binární podobu zprávy. Ke slovům, která se vyskytují na pozicích jedničkových bitů, vložím do XML vlastní tag `<w:rStyle>`, který značí mnou vytvořený styl. Ještě než jej do XML dokumentu vložím, musím odstranit tagy, které by s mým stylem byly v konfliktu. Jedná se o tagy `<w:rFonts>`, `<w:sz>` a `<w:szC>`.

Algoritmus 1 Jednoduchý algoritmus popisující princip vložení tajné zprávy pomocí Baconovy šifry

Vstup: cover soubor .docx/.txt, tajná zráva (MSG)
Výstup zašifrovaný soubor obsahující tajnou zprávu

```
Převedení MSG do binární podoby
Jednotlivé znaky MSG jsou převedeny do ascii hodnoty
Převedeným znakům jsou přiřazeny odpovídající vzory z pole bacons_table
bit = next(MSG)
while bit do
    for slovo in cover text do
        if bit == 1 then
            nahraj vlastní XML styl
        end if
        bit = next(MSG)
    end for
end while
```

[4]Některé tahy tvoří příčné ukončení. Mají takzvanou patku.

```
▼<w:r>
  ▼<w:rPr>
     <w:rStyle w:val="baconstyle"/>
   </w:rPr>
   <w:t xml:space="preserve">Ukázka </w:t>
  </w:r>
```

Obrázek 4.6: Styl použitý pro Baconovu šifru.

Dešifrování

Dešifrování Baconovy šifry probíhá ve funkci `bacon_decode` v souboru `bacon.py`. Postupně procházím slova jednotlivých běhů[5] a kontroluji, jaký mají nastavený styl.

```
1        if run.style.name == "baconstyle":
2            binary += '1';
3        elif run:
4            if run.text != ' ':
5                binary += '0';
```

Výpis 4.1: Dešifrování ukryté zprávy Baconovou šifrou.

Poté, co jsem prošel celý zašifrovaný text, jsem získal binární podobu ukryté zprávy. Tu musím rozdělit do skupinek po 5 bitech (Baconova šifra je 5bitové binární kódování) a každou skupinku převést na odpovídající znak. To probíhá ve funkci `bacon_pattern_to_string`.

```
bacons_table = ["00000", "00001", "00010", "00011", "00100", "00101",
                "00110","00111","01000","01001","01010","01011","01100",
                "01101","01110","01111","10000","10001","10010","10011",
                "10100", "10101","10110", "10111"]

alphabet     = ["A", "B", "C", "D", "E", "F", "G", "H", "(I,J)", "K", "L",
                "M", "N", "O", "P","Q","R", "S", "T", "(U/V)", "W", "X",
                "Y", "Z"]
```

```
1    bacons_decoded_message = ""
2        for k in range(len(bacons_patterns)):
3            for l in range(len(bacons_table)):
4                if(bacons_patterns[k] == bacons_table[l]):
5                    bacons_decoded_message += alphabet[l]
```

Výpis 4.2: Převod binární sekvence na čitelnou podobu.

`Bacons_patters` jsou již vytvořené binární skupiny. Jednotlivé skupiny postupně testuji, zdali odpovídají některému ze vzorů z `bacons_table`, následně uložím do zprávy znakovou podobu tohoto vzoru.

Z důvodu rychlosti jsem tuto metodu ještě pozměnil. Metodu jsem rozšířil o terminační znak `"." = "11111"` představující ukončení zprávy. Pokud tento znak není zadán, je dešifrován celý soubor.

[5] běh je ve fromátu docx sekvence znaků se stejným stylem, může to být odstavec, zvýrazněné slovo a další

4.2.4 Implementace – Open-space metoda

Open-space metoda (viz 2.4.1), podobně jako Baconova šifra, spočívá ve formátování textu. Jako variantu této metody jsem zvolil mezislovní mezery. Tuto metodu jsem ještě upravil, aby dosahovala lepších výsledků. Místo přidání mezery jen zvětším velikost mezery stávající. Díky této úpravě by měla být metoda mnohem méně postřehnutelná (viz 5.3).

Šifrování

Stejně jako u Baconovy šifry i zde pracuji s XML podobou dokumentu. Ve funkci `create_whitespace_el` v souboru `whitespaces.py` vytvářím XML element `<w:sz>`, kterému nastavuji hodnotu na `19`. Tato hodnota odpovídá velikosti mezery `9.5`. Velikost mezery jsem záměrně zvolil takovou, aby byla minimální šance na zničení tajné zprávy a zpráva byla zároveň nepostřehnutelná.

Velikost mezery je vždy totožná s velikostí textu, nejčastěji používaná velikost textu je 11. Rozdíl mezi velikosti mezery 9.5 a 11 je minimální. Stejně tak šance, že by uživatel použil velikost textu 9.5, což by znemožnilo použití této metody s použitými parametry, je velmi nepravděpodobná.

Opět procházím XML soubor a jednotlivé elementy, včetně stylu, kopíruji do nového XML souboru. Při šifrování touto metodou nás zajímají pouze mezery. Mezeru v XML struktuře poznáme tak, že je prázdný text element. To znamená, že tag `<w:t>` neobsahuje žádný text. Pokud se takový element nachází na pozici odpovídající bitu “1” (opět na základě binární podoby zprávy, kterou si zvolí uživatel), vložím do něj nový element značící velikost mezery `9.5`.

```
▼<w:r>
  ▼<w:rPr>
     <w:rFonts w:ascii="Times New Roman" w:cs="Times New Roman" w:hAnsi="Times New Roman"/>
   </w:rPr>
   <w:t xml:space="preserve"> </w:t>
  </w:r>
▼<w:r>


▼<w:r>
  ▼<w:rPr>
     <w:rFonts w:ascii="Times New Roman" w:cs="Times New Roman" w:hAnsi="Times New Roman"/>
     <w:rStyle w:val="spaces_style"/>
     <w:sz w:val="19"/>
   </w:rPr>
   <w:t xml:space="preserve"> </w:t>
  </w:r>
```

Obrázek 4.7: První blok kódu reprezentuje mezeru beze stylu, druhý blok reprezentuje mezeru, které jsem přiřadil nový element.

Algoritmus 2 Jednoduchý algoritmus popisující princip vložení tajné zprávy pomocí Open-space metody

Vstup: cover soubor .docx/.txt (C), tajná zráva (MSG)
Výstup zašifrovaný soubor obsahující tajnou zprávu

```
1: Převedení MSG do binární podoby
2: bit = next(MSG)
3: while bit do
4:     slova = cover_text.rozdělit_na_slova("1+ bílých znaků")
5:     for slovo in slova do
6:         if slovo == ' ' then
7:             if bit == '1' then
8:                 změna velikosti mezery
9:             end if
10:            bit = next(MSG)
11:        end if
12:    end for
13: end while
```

Dešifrování

Dešifrování probíhá ve funkci `spaces_decode`. K dešifrování jsem využil funkcí knihovny `docx`. Knihovna je schopná zjistit styl jednotlivých běhů, ale není schopná zjistit jejich velikost. Z toho důvodu jsem do elementů mezer přidal prázdný styl `spaces_style`, který neobsahuje žádné vlastnosti a slouží tedy pouze jako flag, značící bity "1".

Procházím jednotlivé slova dokumentu a zjišťuji jejich styl. Mezeru se stylem `spaces_style` (tedy mezeru velikosti 9.5) převedu na bit "1" , ostatní mezery na bit "0".

```
1  for paragraph in doc.paragraphs:
2      for run in paragraph.runs:
3
4          if run.style.name == "spaces_style":
5              binary = binary + '1';
6          elif run.text == " ":
7              binary = binary + '0';
```

Výpis 4.3: Dešifrování ukryté zprávy Open space metodou.

Získám tedy ukrytou zprávu v binární podobě, tu jen snadno převedu na textovou podobu pomocí funkce `binary_to_str`. Kódování pro tuto metodu jsem zvolil rozšířené ASCII, to znamená, že každý znak je nutno kódovat 8 bity.

4.2.5 Implementace – metoda synonym

Tato metoda (viz 2.4.2) se od předchozích liší ve způsobu ukrytí zprávy, neřadí se mezi metody založené na formátování textu, nýbrž mezi metody lingvistické.

Šifrování

Pro účely ukrytí tajné zprávy v binární podobě, jsem nejdříve potřeboval vytvořit slovníky anglických slov a jejich synonym[6]. Slovníky jsou globální a nachází se v souboru `synonyms.py`.

```
dictionary_of_zeros = OrderedDict((("smart",{0}),("dry",{0})...))
dictionary_of_synonyms = OrderedDict((("wise",{1}),("arid",{1})...))
```

Oba slovníky obsahují okolo 150 slov. Jeden slovník obsahuje často používaná anglická slova, které budu v cover textu hledat a druhý slovník obsahuje synonyma k těmto slovům, kterými je budu nahrazovat.

Nejprve si ověřím, že zvolený soubor má dostatečnou kapacitu na ukrytí zvolené tajné zprávy. Jelikož k ukrytí používám 8-bitové kódování, ověřuji tuto vlastnost následovně. Délka zprávy vynásobená 8 musí být menší nebo rovna počtu slovníkových slov, nacházejících se v textu. Ty zjistím pomocí funkce `count_dictionary_words`.

Ještě než se dostanu k samotnému nahrazování slov, je potřeba zjistit s jakým bitem zprávy momentálně pracuji.

```
1   for word in words:
2       if cnt >= message_len:
3           break
4       else:
5           if word.lower() in synonyms.dictionary_of_zeros:
6               cnt += 1
7               bit = next(msg_iter, None)
8           elif word.lower() in synonyms.dictionary_of_synonyms:
9               cnt += 1
10              bit = next(msg_iter, None)
11          else:
12              bit = "x"
13      new_run = new_run_element(word, bit, run_props, method)
14      new_runs.append(new_run)
```

Výpis 4.4: Šifrování metodou synonym.

Postupně procházím jednotlivá slova každého běhu a zjišťuji bit zprávy. Následně volám funkci `new_run_element`, ve které provádím samotné nahrazování slov. Zároveň v tomto cyklu navyšuji počítadlo `cnt` vždy, když naleznu slovo nacházející se v některém ze slovníků. Počítadlo je zde z toho důvodu, abych šifroval minimální počet slov potřebný k ukrytí celé tajné zprávy.

Z cyklu se přesouvám do zmíněné funkce `new_run_element`. Každé nalezené slovo použiji k ukrytí jednoho bitu tajné zprávy.

Pokud se nalezené slovo vyskytuje v prvním slovníku (`dictionary_of_zeros`) a potřebuji ukrýt jedničkový bit, nahradím jej za slovo podobného významu z druhého slovníku (`dictionary_of_synonyms`). Podobně je tomu v případě, že se nalezené slovo nachází ve slovníku synonym a potřebuji ukrýt nulový bit. V tomto případě jej naopak zaměním za slovo z původního slovníku (dictionary_of_zeros). Jestliže je nalezené slovo z prvního slovníku a potřebuji ukrýt nulový bit, nebo je ze slovníku druhého a potřebuji ukrýt jedničkový bit, neprovádím žádnou operaci. Tento postup je znázorněn v následující ukázce kódu.

[6]Slova stejného, nebo podobného významu

```
    for word in text:
        text_el = xml.etree.ElementTree.Element(text_tag)
        if(word.lower() in dictionary_of_zeros):
            index = list(dictionary_of_zeros.keys()).index(word.lower())
            #jednickove bity nahradim jejich synonymem
            if bit == '1':
                #nalezeni prislusneho synonyma ve druhem slovniku, podle indexu
                syn_word = list(dictionary_of_synonyms.keys())[index]
        elif(word.lower() in dictionary_of_synonyms):
            index = list(dictionary_of_synonyms.keys()).index(word.lower())
            if bit == '0':
                syn_word = list(dictionary_of_zeros.keys())[index]
        text_el.text = syn_word
```

Výpis 4.5: Nahrazování slov metodou synonym.

Algoritmus 3 Jednoduchý algoritmus popisující princip vložení tajné zprávy pomocí metody synonym

Vstup: cover soubor .docx/.txt (C), tajná zráva (MSG)
Výstup zašifrovaný soubor obsahující tajnou zprávu

```
Převedení MSG do binární podoby
bit = next(MSG)
cnt = 0
while bit do
    for slovo in cover text do
        if cnt >= len(MSG) then
            break
        else
            if slovo in slovniky then
                cnt += 1
                if bit == 1 and slovo in slovnik puvodnich slov then
                    nahrazení za příslušné synonymum
                else if bit == 0 and slovo in slovnik synonym then
                    nahrazení za příslušné původní slovo
                end if
                bit = next(MSG)
            end if
        end if
    end for
end while
```

Dešifrování

Dešifrování touto metodou je velmi jednoduché. Při dešifrování procházím textem původního cover souboru a dělím jej na jednotlivá slova. Pokud se slovo nachází ve slovníku synonym reprezentuje bit “1” . Pokud se slovo nachází ve druhém slovníku, značí bit “0”.

```
    for paragraph in doc.paragraphs:
        text = text + paragraph.text + " "

        for run in paragraph.runs:
            t = run.text
            split = t.strip().split(" ")

            for word in split:
                word = word.lower()
                if ((word in dictionary_of_synonyms):
                    binary += "1"
                elif word in dictionary_of_zeros:
                    binary += "0"

    secret_message = steganography.binary_to_str(binary)
    return secret_message
```

Výpis 4.6: Dešifrování metodou synonym.

4.2.6 Vlastní metoda 1

Při implementaci vlastních metod jsem vycházel jednak z původního návrhu, především však z výsledků testů. Testy ukázaly, že metoda synonym má velký steganografický potenciál. Je málo nápadná a navíc robustní vůči formátovacím změnám. Proto jsem se rozhodl vycházet z této metody, kterou jsem modifikoval pro zlepšení jejích vlastností.

Upravená metoda tedy namísto původního 8bitové kódování ukrývá zprávu již známým 5bitovým Baconovým šifrováním.

Princip ukrytí je velmi podobný jako u metody synonym. S tím rozdílem, že tajná zpráva nejprve převedena do jednotlivých vzorů z Baconovy tabulky, která se nachází v souboru `bacon.py`.

Tajnou zprávu při dešifrování získám převedením jednotlivých slov ze slovníků do jejich binární podoby. Binární podobu tajné zprávy nakonec dělím na posloupnosti pěti znaků, kterým přiřazuji odpovídající textovou podobu.

```
    if(method == "own1"):
        bacons_patterns = re.findall('.....',binary)
        secret_message = bacon.bacon_pattern_to_string(bacons_patterns, bacon.bacons_table)
```

Výpis 4.7: Převod binární zprávy do čitelné podoby s využitím Baconovy tabulky.

4.2.7 Vlastní metoda 2

Druhá metoda obsahuje generování Huffmanova stromu, jehož princip jsem vysvětlil v návrhu metod (viz 3.4.2).

Nejprve si zjistím četnost jednotlivých znaků tajné zprávy, a následně generuji Huffmanův strom. Samotné generování stromu jsem převzal[7], neboť se jedná o složitý algoritmus, který navíc není hlavním tématem této práce.

Po vygenerování stromu má každý znak na základě četnosti přidělen vlastní binární podobu. Z těchto vzorů je následně složena binární zpráva, která je principem metody synonym (viz 4.4) vložena do cover souboru.

[7] Kód převzat z této stránky: https://www.programiz.com/dsa/huffman-coding

4.2.8 Automatizované testy

Testování probíhá v souboru `tests.py`. Po spuštění tohoto skriptu se postupně vykonávají jednotlivé testy nacházející se ve funkci `main`.

```
1    #sifrovani vsech souboru ve slozce cover_files
2    encode_all_covers()
3    #desifrovani a kontrola, zda zustala zachovana tajna zprava
4    decode_all_encodes()
5    #zmena formatovani a kontrola zmeny tajne zpravy
6    check_robustness()
7    #vypocet zmeny velikosti souboru po vlozeni tajne zpravy
8    calculate_SIR()
9    #vykresleni grafu
10   plot_all()
```

Výpis 4.8: Jednotlivé testy

```
BACON ENCODING:
1 encoded ✓ (adele.docx)
2 encoded ✓ (al-green.docx)
3 encoded ✓ (alicia-keys.docx)
4 encoded ✓ (amy-winehouse.docx)
5 encoded ✓ (beatles.docx)
6 encoded ✓ (bieber.docx)
...
23 encoded ✓ (romeo_and_juliet.docx)
24 encoded ✓ (The_little_prince.docx)
Elapsed time:      18.901144 [seconds]
Cover texts size:  1667432   [bytes]
Efficiency:        0.011335  [time in seconds to encode 1KB]
Average capacity:  2.760508  [bits]
-------------------------------
Success rate:   24/24 = 100%
-------------------------------
BACON SIR:      4.801248%

BACON DECODING:
All messages DECODED SUCCESSFULLY!

BACON ROBBUSTNESS CHECK:
After FONT CHANGES 0/24 messages sucesfully decoded
```

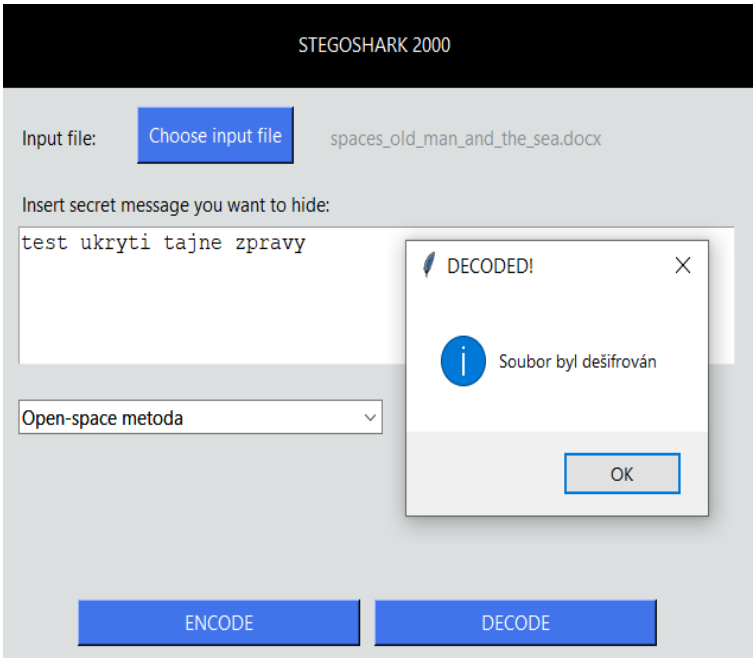
Obrázek 4.8: Výstup testu

Na výstupu můžeme vidět výsledky všech testů. Rychlost metody, čas potřebný k zašifrování 1KB dat, průměrnou kapacitu a její úspěšnost, celkový nárůst velikosti souboru (počítán jen z úspěšně zašifrovaných souborů) a nakonec kolik ukrytých zpráv bylo zachováno i po formátovacích změnách souboru.

Součástí testů je i vytvoření grafů, které se nachází v následující kapitole.

4.2.9 Uživatelské rozhraní

Uživatelské rozhraní je implementováno v souboru `GUI.py`. K jeho vytvoření jsem použil knihovnu `tkinter`.



Obrázek 4.9: Uživatelské rozhraní pro snadné šifrování a dešifrování steganografickými metodami.

Nejprve si uživatel zvolí vstupní soubor. Kliknutím na tlačítko "Choose input file" se zavolá funkce `Choose input`, ve které pomocí funkce knihovny Tkinter `filedialog.askopenfile` zvolíme požadovaný soubor.

Dále se zde nachází kombinované pole (combo box), kterým si uživatel zvolí steganografickou metodu.

```
1    combo = ttk.Combobox(root,textvariable=selected_method, width=35)
2    combo['values'] = ['Baconova sifra', 'Open-space metoda', 'Metoda synonym', 'Metoda synonym s
         baconovzm sifrovanim', 'Huffmanovo kodovani']
3    combo['state'] = 'readonly'
4    combo.set('---')
```

Výpis 4.9: Vytvoření kombinovaného pole.

Změnu hodnoty kombinovaného pole sleduji pomocí funkce `bind`. Získání jeho hodnoty probíhá ve funkci `method_changed`.

Textové pole pro vstupní zprávu je implementováno pomocí prvku `Text` a zisk jeho hodnoty zajišťuje funkce `get`.

Jakmile si uživatel zvolí všechny povinné možnosti šifrování, potvrdí zašifrování tlačítkem `ENCODE`. Stisknutí vyvolá stejnojmennou funkci, která zpracuje zadané vstupní požadavky a provede samotné zašifrování.

```
    secret_mes = message.get('1.0', 'end-1c')
    method = combo.current()
    #filepath je cesta k souboru ktery si zvolil uzivatel
    filepath = os.path.abspath(inputfile.name)
    if method == 0:
        method_name = '-b'
    elif method == 1:
        method_name = '-w'
    elif method == 2:
        method_name = '-r'
    elif method == 3:
        method_name = '--own1'
    elif method == 4:
        method_name = '--own2'
    #samotne sifrovani
    steganography.main(['-i', str(filepath), '-e', '-s', secret_mes, method_name])
```

Výpis 4.10: Zpracování argumentů a následné šifrování.

Dešifrování probíhá stejným způsobem, ve funkci `decode`.

Do finálního GUI jsem oproti návrhu nedal možnost výběru lokace pro uložení výsledku. Při šifrování se ve složce se zdrojovými kódy vytvoří nová složka `/encoded`, ve které se budou nacházet všechny uživatelem zašifrované soubory. To samé platí pro dešifrování. Taktéž je vytvořená nová složka `/decoded`.

Kapitola 5

Testování a analýza vybraných metod

5.1 Testovací data

Pro testování implementovaných metod jsem vytvořil rozsáhlý dataset textových souborů. Některé soubory jsem vytvořil, jiné převzal [15, 17, 18]. Soubory jsou různorodé, při testování používám více textových formátů, různé fonty písma a formátování.

5.2 Úspěšnost implementovaných metod

Testování jsem prováděl na vlastních benchmarkových testech. Zaměřil jsem se na tyto vlastnosti metod: `rychlost metody, úspěšnost ukrytí zprávy, maximální velikost zprávy vzhledem k velikosti textu, změna velikosti souboru, robustnost a postřehnutelnost`.

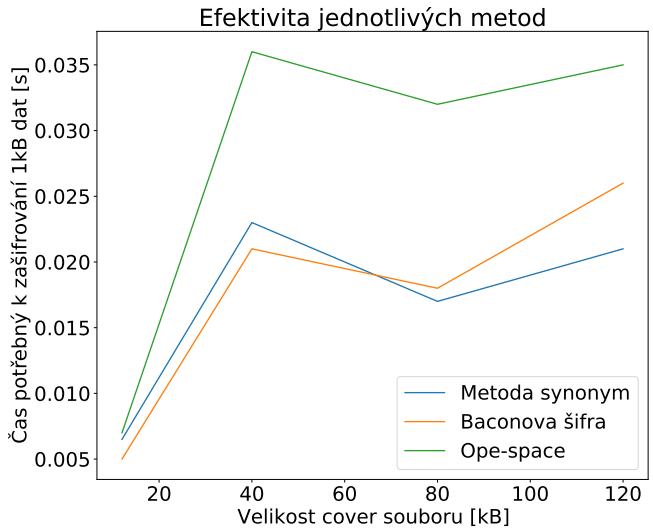
5.2.1 Rychlost a efektivita metod

V testu rychlosti zvítězila metoda synonym, která je ze všech implementovaných metod nejrychlejší a tedy i nejefektivnější. Metoda na rozdíl od ostatních nepracuje s XML reprezentací dokumentu a pouze provádí čtení ze slovníků, což je časově nenáročná operace.

Průměrná efektivita [s]	Baconova šifra	Metoda synonym	Open-space metoda
Krátká zpráva	0.011632	0.010929	0.016486
Dlouhá zpráva	0.012854	0.010577	0.018910

Tabulka 5.1: Čas potřebný k ukrytí jednoho kilobajtu dat u jednotlivých metod.

Na následujícím grafu je znázorněna efektivita metod vůči velikosti vstupního souboru. Efektivitu počítám jako čas potřebný k zašifrování 1KB dat.



Obrázek 5.1: Efektivita metod vzhledem k velikosti souboru.

5.2.2 Úspěšnost ukrytí zprávy

Pokud cover text nemá dostatek slov na ukrytí celé zprávy, je ukrytí považováno za neúspěšné.

Při testování nejmenší úspěšnost zaznamenala metoda synonym. Pro ukrytí tajné zprávy touto metodou je zapotřebí obrovské množství anglického textu (nejlépe nějaké knihy, články, texty písní atd.).
Nejlépe v tomto testu skončila Baconova metoda, která k zašifrování informace potřebuje pouhých 5 bitů.

Úspěšnost ukrytí [%]	Baconova šifra	Open-space metoda	Metoda synonym
Krátká zpráva (do 10 znaků)	≈ 100	94,444	68.4211
Dlouhá zpráva (do 50)	70.5882	67.6471	42.8571
Velmi dlouhá zpráva (nad 100)	61.9048	59.5238	14.2857

Tabulka 5.2: Úspěšnost ukrytí tajné zprávy u jednotlivých metod.

Z tabulky lze vyčíst, že je metoda synonym téměř nepoužitelná pro ukrytí delších zpráv a lze ji spolehlivě použít jen pro opravdu krátké zprávy. Zbylé metody mají nadpoloviční úspěšnost i u delších zpráv.

5.2.3 SIR (Size Increasing Ratio)

SIR nám udává, o kolik je nově vzniklý zašifrovaný soubor větší, než soubor původní.

SIR = (stego.docx size - cover.docx size)/ cover.docx size * 100 [4]

Tato steganografická vlastnost je počítána pouze z úspěšně zašifrovaných souborů. Pokud by testy obsahovaly soubory, které by nebyly úspěšně zašifrované, SIR by vycházelo chybně (pravděpodobně záporné, nebo velmi malé).

Average Size Increasing Ratio [%]	
Baconova šifra	19,638
Open-space method	57,07
Metoda synonym	22,722

Tabulka 5.3: Procentuální navýšení velikosti nově vzniklého souboru.

Největší nárůst velikosti souboru zaznamenala Open-space metoda, která navyšuje velikost souboru o více než 50 % . Pokud bychom tedy chtěli šifrovat soubory o velkém objemu dat, bylo by vhodné zvolit jednu ze zbylých 2 metod.

5.2.4 Kapacita metod

Za předpokladu, že jeden znak zabírá 1 bajt v paměti jsem spočítal procento kapacity, jako poměr kapacity (viz 2.5.3), vynásobený 100.

Metody	Cover text [bytes]	Počet slov (v cover textu)	Maximální počet bitů, který lze ukrýt	Počet znaků, které lze metodou ukrýt	Kapacita [%]
Baconova šifra	46615	18975	18975	4033	**8,004**
	81768	26700	26700	5300	
Open-space metoda	46615	18975	18975	2520	**5,211**
	81768	26700	26700	3312	
Metoda synonym	46615	18975	662	82	**0,158**
	81768	26700	1094	136	

Obrázek 5.2: Výsledná kapacita je spočítána pro všechny testovací soubory, ne pouze pro soubory uvedené v tabulce.

5.2.5 Robustnost

Testy na robustnost jsem rozdělil na 3 menší testy, z nichž každý testuje jiný způsob formátování textu. Testy provádí `změnu velikosti písma, změnu fontu písma` a nakonec `všechny možné formátovací změny` zároveň.

Operace	Změna fontu písma		
Metoda	Baconova šifra	Open-space metoda	Metoda synonym
Výsledek	Zpráva je zničena	Zpráva se zachová	Zpráva se zachová

Tabulka 5.4: Důsledky změny fontu pro celý dokument.

Změna fontu písma zničí tajnou zprávu ukrytou Baconou šifrou, která k ukrytí používá právě tuto formátovací vlastnost.

Operace	Změna velikosti písma		
Metoda	Baconova šifra	Open-space metoda	Metoda synonym
Výsledek	Zpráva se zachová	Zpráva je zničena	Zpráva se zachová

Tabulka 5.5: Důsledky změny velikosti písma pro celý dokument.

Operace	Provedení všech formátovacích úprav najednou		
Metoda	Baconova šifra	Open-space metoda	Metoda synonym
Výsledek	Zpráva je zničena	Zpráva je zničena	**Zpráva se zachová**

Tabulka 5.6: Všechny formátovací změny najednou (změna fontu, velikosti písma, odřádkování, zarovnání textu).

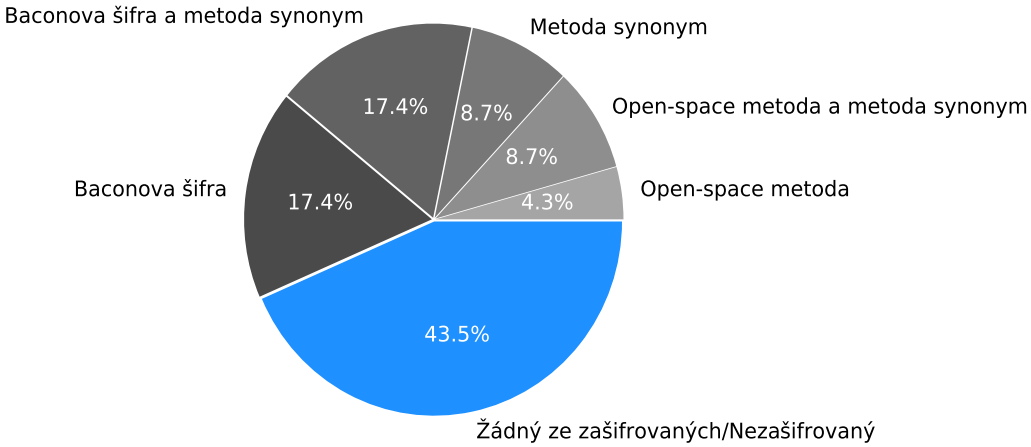
Na základě těchto dat lze s jistotou říci, že nejrobustnější metoda je `metoda synonym`. Zpráva ukryta touto metodou byla zachována ve 100 % případů. Zpráva by mohla být porušena pouze v případě, že by uživatel ze zašifrovaného souboru smazal, nebo naopak přidal některé ze slovníkových slov (navíc pouze v dosahu tajné zprávy, tzn. na začátku textu).

5.2.6 Postřehnutelnost

U steganografických metod je nejdůležitější vlastností postřehnutelnost. Na tuto vlastnost je velmi obtížné napsat jakékoliv testy. Rozhodl jsem se tedy jednotlivé texty obsahující ukrytou zprávu, zašifrovanou určitou metodou, poskytnout většímu množství nezaujatých pozorovatelů.

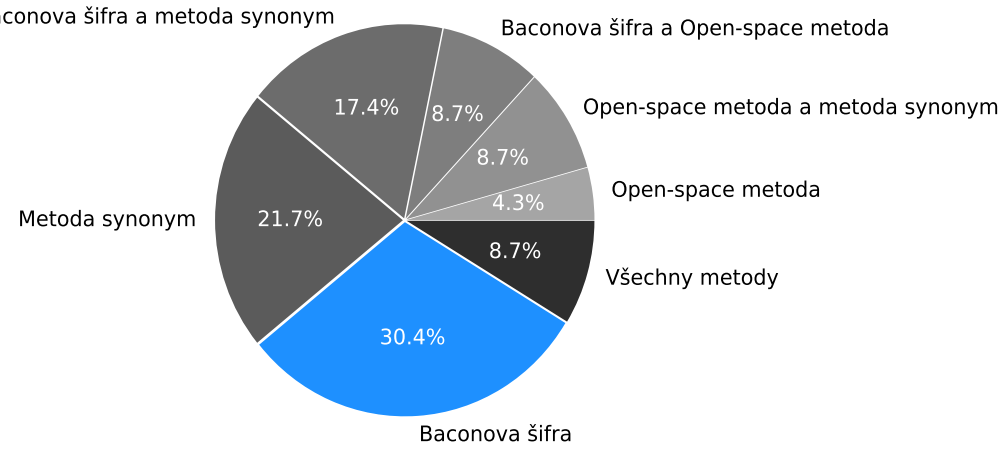
Zajímalo mě, který text na ně působí neupraveně, a který naopak nejvíce nápadně. Výsledky byly přeneseny do grafu.

Metodami v grafu jsou myšleny texty, zašifrované těmito metodami. Dotázaných bylo celkem 23 osob.



Obrázek 5.3: Výsledky dotazu na otázku: “Přijde vám některý ze souborů podezřelý?”.

Při prvním dotazu 10/23 osob (43.5 %) uvedlo, že jim žádný soubor nepřijde podezřelý, nebo označili některý z nešifrovaných souborů. Nikdo nebyl schopný správně označit všechny šifrované. 13/23 osob (56.5 %) správně označilo alespoň 1 šifrovaný dokument.



Obrázek 5.4: Výsledky dotazu na otázku: "Dokážete říci, který soubor je zašifrovaný?".

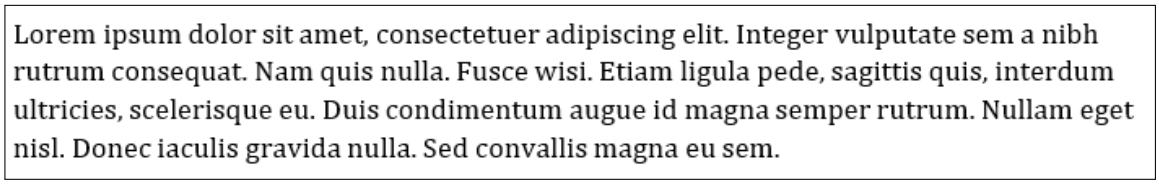
Stejné soubory jsem poskytl i k druhému, podrobnějšímu zkoumání, poté co jsem testovací subjekty ujistil, že se mezi soubory nachází minimálně 1 šifrovaný soubor. Alespoň 1 z nich v tomto případě odhalili všichni, nicméně všechny šifrované soubory správně odhalili pouze 2 osoby (8,7 %).

Ukázky zašifrovaných souborů jednotlivými metodami

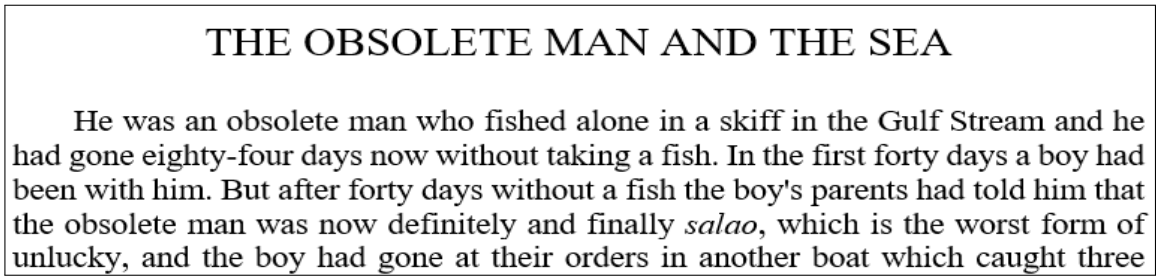
Lorem ipsum dolor sit amet , consectetuer adipiscing elit . Integer vulputate sem a nibh rutrum consequat . Nam quis nulla . Fusce wisi . Etiam ligula pede , sagittis quis , interdum ultricies , scelerisque eu . Duis condimentum augue id magna semper rutrum . Nullam eget nisl . Donec iaculis gravida nulla . Sed convallis magna eu sem. Duis bibendum , lectus ut viverra rhoncus , dolor nunc faucibus libero, eget facilisis enim ipsum id lacus . Sed elit dui, pellentesque a, faucibus vel, interdum nec, diam.

Lorem **ipsum dolor** sit **amet** , **consectetuer** adipiscing **elit** . Integer **vulputate sem a** nibh rutrum **consequat** . Nam **quis** nulla . **Fusce** wisi . **Etiam** ligula **pede** , sagittis quis , **interdum ultricies** , scelerisque **eu** . Duis **condimentum** augue **id magna** semper **rutrum** . Nullam **eget nisl** . Donec **iaculis** gravida nulla . **Sed** convallis magna eu **sem.** Duis **bibendum** , **lectus ut viverra rhoncus** , **dolor nunc** faucibus **libero**, **eget** facilisis enim ipsum **id** lacus . **Sed elit dui**, **pellentesque a**, **faucibus vel**, **interdum nec** , **diam.**

Obrázek 5.5: Text zašifrovaný Baconovou šifrou. Nalevo ideální cover text psaný fontem Times New Roman. Napravo je nepozorované použití metody znemožněno zvýrazněným textem.



Obrázek 5.6: Open-space metoda je téměř nepostřehnutelná.



Obrázek 5.7: Postřehnutelnost u metody synonym záleží na čtenářově zdatnosti v anglickém jazyce.

5.3 Výsledky vlastních metod

5.3.1 Metoda synonym + Baconova šifra

Open-space metoda i metoda synonym dosahovaly nižších hodnot kapacity, než Baconova šifra. Kapacita by šla navýšit jiným typem kódování. V rámci testů jsem se rozhodl tuto hypotézu ověřit. Vytvořil jsem vlastní metodu, která kombinuje hlavní přednosti metody synonym a Baconovy šifry.

Metodu synonym jsem tedy upravil tak, aby byla informace kódována 5 bity, namísto původních 8.

Úspěšnost ukrytí [%]	Metoda synonym (8-bit)	Metoda synonym + Baconovo šifrování (8-bit)
Krátká zpráva (do 10 znaků)	68,4211	78,5714
Dlouhá zpráva (do 50)	42.8571	71.4286
Velmi dlouhá zpráva (nad 100)	14.2857	57.1429
Kapacita	0,158	0,2631

Tabulka 5.7: Úspěšnost ukrytí po použití Baconovy šifry v kombinaci s metodou synonym.

Z výsledků lze vidět, že nově vzniklá metoda zaznamenala výrazné zlepšení z hlediska kapacity a tím pádem i úspěšnosti ukrytí tajné zprávy.

5.3.2 Metoda synonym + Huffmanovo kódování

Jako další implementovanou modifikaci jsem testoval metodu synonym využívající Huffmanovo kódování. Huffmanovo kódování jsem nejprve testoval na vstupních zprávách, u kterých byl výskyt jednotlivých znaků zcela náhodný.

Počet bajtů potřebných k ukrytí tajné zprávy			
Tajná zpráva	Metoda synonym	MS + BŠ	MS + Huffman
zprava1 6[B]	48	30	14
zprava2 20[B]	160	100	86
zprava3 50[B]	400	250	218
zprava4 [200B]	1600	1000	891

Tabulka 5.8: V tabulce 1B dat představuje 1 libovolný znak.

Z tabulky je patrné, že šifrování dat Huffmanovým kódováním v porovnání s původní metodou synonym (využívá rozšířené ascii 8-bit) a Baconovým šifrováním (5-bit) potřebuje k ukrytí tajné zprávy nejmenší množství cover textu.

Toto šifrování nám tedy zaručí nejlepší možnou kapacitu ze všech zkoumaných způsobů ukrytí zprávy.

Pokud by vstupní zpráva navíc obsahovala pouze omezený počet znaků a jejich výskyt by byl tedy procentuálně vyšší, dosáhla by metoda ještě lepších výsledků. Viz následující tabulka.

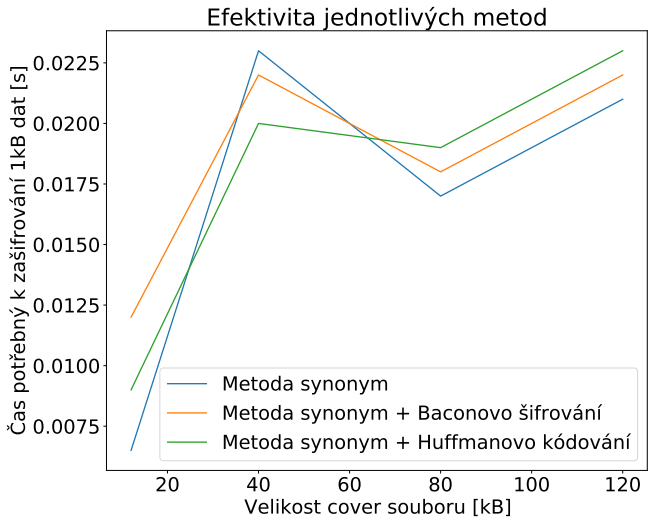
Zpráva 30[B]	“Huffman coding incoming uh oh”
Metoda synonym + Baconovo šifrování	K ukrytí je potřeba 150 bajtů cover textu
Metoda synonym + Huffmanovo kódování	K ukrytí je potřeba pouhých 100 bajtů cover textu

Tabulka 5.9: Pokud je zpráva omezena na menší počet znaků (které se navíc často opakují), lze očekávat výrazné zlepšení z hlediska kapacity.

Znak	""	n	h	o	i	u	f	m	c	g	a
Četnost	4	4	3	3	3	2	2	2	2	2	1
Vzor	100	101	010	001	000	0111	0110	1101	1100	1111	11101

Tabulka 5.10: Vygenerované vzory [16]. Znaku s častějším výskytem je přiřazen kratší vzor.

Hlavní nevýhodou této metody je, že je potřeba znát předem odpovídající Huffmanův strom. Pokud není předem znám, je potřeba jej vložit do cover textu spolu s tajnou zprávou. Zlepšení kapacity se tedy projeví až od určité velikosti tajné zprávy.



Obrázek 5.8: Efektivita vlastních metod vzhledem k velikosti souboru.

Z grafu je jasně patrné, že ačkoli měly změny pozitivní vliv na kapacitu a úspěšnost ukrytí, celková efektivita vůči původní metodě synonym zůstala téměř stejná.

5.4 Zhodnocení výsledků a porovnání metod

V této kapitole se budu věnovat celkovému srovnání všech implementovaných metod. Metody hodnotím na základě kapitol "Úspěšnost implementovaných metod" (viz 5.2) a "Výsledky vlastních metod" (viz 5.3), ve kterých jsem napříč provedenými epxerimenty zhodnotil všechny důležité steganografické vlastnosti jednotlivých metod.

Steganografické vlastnosti metod hodnotím na škále 1 (nejlepší) - 3 (nejhorší).

Metoda	Úspěšnost	Kapacita	Robustnost	Postřehnutelnost
Baconova šifra	**1**	**1**	3	2-3
Open-space	2	2	2	**1**
Metoda synonym	3	3	**1**	2-3

Tabulka 5.11: Srovnání implementovaných metod.

Výsledky testování potvrdily téměř všechny vlastnosti metod, které jsem na začátku odhadoval z existujících publikací (viz 2.5). Některé se neshodují zcela, například postřehnutelnost Baconovy šifry dle mých výsledků nevyšla jako nejlepší, jak slibovaly publikace, ale řadí se téměř na konec žebříčku.

Na následující tabulce demonstruji výsledky mých upravených metod. V tomto srovnání vyšla původní metoda jako nejhorší ve všech možných aspektech (viz 5.3).

Metoda	Úspěšnost	Kapacita	Robustnost	Postřehnutelnost
Metoda synonym	3	3	stejná	3
Metoda synonym + BŠ	2	2	stejná	2
Metoda synonym + HK	**1**	**1**	stejná	**1**

Tabulka 5.12: Srovnání steganografické metody z pohledu šifrování.

Postřehnutelnost by měla zůstat stejná, neboť se nemění způsob ukrytí informace. Teoreticky ale zaměňujeme menší počet slov, a tak se šance, že čtenář nabude podezření, alespoň trochu snižuje.

Na začátku této práce jsem zmínil, že postřehnutelnost je nejdůležitější steganografickou vlastností. Pokud by nám tedy záleželo pouze na této vlastnosti, která nám zaručí bezpečné doručení tajné zprávy, jednoznačně zvítězila upravená Open-space metoda, využívající formátování mezislovních mezer.

Metody	Výhody	Nevýhody
Baconova šifra	Vysoká kapacita a úspěšnost ukrytí.	Není odolná vůči žádným formátovacím změnám.
Open-space	Nejvyšší neviditelnost. Dobrá kapacita i úspěšnost ukrytí.	Není odolná vůči všem formátovacím změnám.
Metoda synonym	Pokud čtenář nevyniká v anglickém jazyce, je metoda špatně postřehnutelná. Nejvíce odolná vůči jakýmkoliv formátovacím změnám.	Vysoká složitost, z důvodu použití rozsáhlých slovníků. Může docházet ke změně významu původního dokumentu.
Metoda synonym Baconova šifra	Zachovává robustnost jakou poskytuje metoda synonym. Lehce navyšuje kapacitu.	Stejné jako u metody synonym.
Metoda synonym Huffmanovo kódování	Zachovává vysokou robustnost. Výrazně zvyšuje kapacitu a celkovou úspěšnost ukrytí.	Je zapotřebí předem znát konkrétní Huffmanův strom, nebo jej poslat spolu se skrytou zprávou.

Tabulka 5.13: Srovnání hlavních předností metod a jejich nedostatků.

V poslední srovnávací tabulce můžete vidět všechny plusy a mínusy jednotlivých metod. Obecně nelze říci, která metoda je nejlepší, každá má své vhodné využití a každá z nich si dokáže ve steganografickém světě najít své místo.

5.5 Možné modifikace implementovaných metod

Žádná metoda není dokonalá, každá má své silné i slabé stránky. S již implementovanými metodami by šlo samozřejmě manipulovat, různě je mezi sebou kombinovat a snažit se tak docílit ještě lepších výsledků.

V Baconově šifře by bylo ideální měnit font podle fontu okolo šifrovaného slova. Bylo tedy potřeba ke každému fontu najít font alternativní (ten nejpodobnější) a nahrazovat jej tímto. To by zajistilo, že bez ohledu na font původního cover textu by byl nově vzniklý text méně nápadný.

Text šifrovaný metodou synonym, jak jsme již zjistili, je sice nejvíce robustní, ale pro čtenáře s dobrou úrovní angličtiny i dosti nápadný. V anglickém jazyce existuje vždy více slov stejného významu, avšak jejich použití se mění dle kontextu. Příkladem je slovo `silný - strong`, můžeme jej nahradit slovy jako `firm, powerful, potent` a jinými. Powerful lze například použít u osob a lidských vlastností (powerful voice = silný hlas), potent lze použít, pokud chceme říci, že má na nás něco silný efekt (potent drink = silný drink) a firm se nejčastěji pojí s materiálem. Pokud bychom tedy v textu zvolili špatnou kombinaci těchto slov (potent voice, firm drink...), text by působil zcela nesmyslně.

Taktéž existují anglická slova, která mají jiný význam jako podstatné jméno a jiný jako přídavné jméno (play = hra/hrát, workout = trénink/cvičit)

Opět by tedy bylo ideální nahrazovat slova na základě kontextu, přídavná jména nahradit pouze za přídavná a podstatná pouze za podstatná. Dále by bylo potřebné zjistit si slovo následující, zařadit jej do určité skupiny slov a na základě této skupiny správně nahradit slovo s odpovídajícím kontextem.

Kapitola 6

Závěr

Cílem této bakalářské práce bylo se dopodrobna seznámit s nepříliš známou vědní disciplínou spadající do kryptografie, steganografií. Poznat blíže její historii, vývoj, různé oblasti, zejména steganografii pro skrývání informace v textových datech.

Součástí práce bylo navrhnout a implementovat některé metody z oblasti digitální textové steganografie. Úspěšně se mi podařilo implementovat několik existujících metod, včetně 2 vlastních metod. Mé vlastní metody spočívají v kombinaci existujících metod za účelem zlepšení důležitých steganografických vlastností.

Pro vyhodnocení implementovaných metod jsem navrhl a vytvořil automatizované testy, které vyhodnotily hlavní steganografické vlastnosti těchto metod. Výsledky byly přeneseny do tabulek a grafů (viz 5). Z testů se nedá jednoznačně určit, která metoda je ve steganografické oblasti nejdokonalejší. Nicméně nám testy dokáží na základě naší preference poradit vhodnou steganografickou metodu. Jestliže víme, že se souborem bude manipulováno, bezkonkurenční je metoda synonym. Pokud ale chceme zprávu ukrýt nepozorovaně, a víme, že soubor nebude již měněn, je vhodnější Open-space metoda. Pro anglické texty lze použít metodu synonym doplněnou o Huffmanovo kódování, u kterého jsme si dokázali, že poskytuje výrazně vyšší kapacitu pro ukrytí tajné zprávy.

Všechny stanovené cíle této bakalářské práce byly splněny, navíc jsem pro účely snazší demonstrace práce vytvořil jednoduché grafické uživatelské rozhraní, umožňující snadné šifrování a dešifrování jednotlivých souborů.

Na tuto práci by šlo dále navázat například implementováním zmíněných modifikací metod, které jsem popsal vkapitole “Možné modifikace implementovaných metod” (viz 5.5), nebo doplněním GUI o další existující metody z oblasti digitální textové steganografie.

Literatura

[1] Abduallah, W. A Review on Steganography Techniques. *American Scientific Research Journal for Engineering, Technology, and Sciences*. Srpen 2016, sv. 24, s. 131–150, [cit. 2022-04-14]. Dostupné z: https://www.researchgate.net/publication/307168058_A_Review_on_Steganography_Techniques.

[2] Agarwal, M. *Text Steganographic approaches: A Comparison* [online]. Department of Computer Science and Engineering, PDPM-IIITDM, Jabalpur, India, 2013 [cit. 2021-21-12]. Dostupné z: https://arxiv.org/ftp/arxiv/papers/1302/1302.2718.pdf.

[3] Ahvanooey, T., Milad, Li, Qianmu, Hou et al. AITSteg: An Innovative Text Steganography Technique for Hidden Transmission of Text Message via Social Media. IEEE Access. 2018, s. 65981–65995, [cit. 2022-23-03]. Dostupné z: https://www.researchgate.net/publication/327034906_AITSteg_An_Innovative_Text-Steganography_Technique_for_Hidden_Transmission_of_Text_Message_via_Social_Media.

[4] Baawi, S., AL Nasrawi, D. a Abdulameer, L. Improvement of "Text Steganography Based on Unicode of Characters in Multilingual"by Custom Font with Special Properties Improvement of "Text Steganography Based on Unicode of Characters in Multilingual"by Custom Font with Special Properties. *IOP Conference Series Materials Science and Engineering*. Červenec 2020, sv. 870, [cit. 2022-23-03]. DOI: 10.1088/1757-899X/870/1/012125.

[5] Da, A., Wahi, J. S., Anand, M. a Rana, Y. *Multi-Image Steganography Using Deep Neural Networks*. 2021 [cit. 2022-24-03]. Dostupné z: https://arxiv.org/pdf/2101.00350.pdf.

[6] Figueira, J. *A Survey On Semantic Steganography Systems*. Prosinec 2021 [cit. 2022-10-04]. Dostupné z: https://www.researchgate.net/publication/357105085_A_Survey_On_Semantic_Steganography_Systems.

[7] Gathen, J. von zur. Chapter E Steganography. In: *CryptoSchool*. Springer Berlin Heidelberg, 2015, s. 418 [cit. 2022-23-03]. ISBN 978-3-662-48425-8. Dostupné z: https://doi.org/10.1007/978-3-662-48425-8_14.

[8] Henry, W. C. Covert Channels within IRC. In:. 2011 [cit. 2021-23-04]. Dostupné z: https://www.semanticscholar.org/paper/Covert-Channels-within-IRC-Henry/a51ceb499a9796cfc704a3e0b2562a11e8a2daed.

[9] Hološka, J. *Steganografie Jako Hrozba Úniku Kritických Obchodních Informací A Její Detekce Pomocí Umělých Neuronových Sítí* [online]. Trilobit, 2009 [cit.

2022-30-03]. Dostupné z: http://trilobit.fai.utb.cz/steganografie-a-jeji-detekce-pomoci-umelych-neuronovych-siti.

[10] Janušek, R. *Steganografie s využitím neuronových sítí* [online]. Ostravská univerzita v Ostravě, 2015 [cit. 2022-24-03]. Dostupné z: https://konference.osu.cz/svk/sbornik2015/pdf/budoucnost/informatika/Jarusek.pdf.

[11] Juráň, M. *Steganografie*. 2008 [cit. 2021-12-30]. [cit. 2022-23-03]. 71 s. Diplomová práce. Mendelova zemědělská a lesnická univerzita v Brně. Vedoucí práce Foltýnek, T. Dostupné z: https://adoc.pub/steganografie-mendelova-zemdlska-a-lesnicka-univerzita-v-brn.html.

[12] Kipper, G. *Investigator's guide to steganography*. 1. vyd. Auerbach publications, 2004 [cit. 2021-23-11]. 209 s. ISBN 0-8493-2433-5.

[13] Kishore, V., Chen, X., Wang, Y., Li, B. a Weinberger, K. Q. Fixed Neural Network Steganography: Train the images, not the network. In: *International Conference on Learning Representations*. 2022 [cit. 2022-23-03]. Dostupné z: https://openreview.net/forum?id=hcMvApxGSzZ.

[14] Kolla, A. *List of 10 Best Steganography Tools to Hide Data* [online]. Geek Dashboard, 2020 [cit. 2021-21-12]. Dostupné z: https://www.geekdashboard.com/best-steganography-tools/.

[15] Lit, W. V. *Email Spam* [online]. kaggle, 2019 [cit. 2022-30-03]. Dostupné z: https://www.kaggle.com/datasets/veleon/ham-and-spam-dataset.

[16] Ltd., P. L. P. *Huffman Coding* [online]. Programiz, no date [cit. 2022-30-03]. Dostupné z: https://www.programiz.com/dsa/huffman-coding.

[17] Mooney, P. *Song Lyrics* [online]. kaggle, 2018 [cit. 2022-30-03]. Dostupné z: https://www.kaggle.com/datasets/paultimothymooney/poetry.

[18] Mosser, S. *Steganography-dataset* [online]. Github, 2015 [cit. 2021-21-12]. Dostupné z: https://github.com/mosser/steganography-dataset/tree/master/txt.

[19] Rafat, K. F. a Sher, M. *Survey report - state of the art in digital steganography focusing ascii text documents* [online]. Department of Computer Science International Islamic University Islamabad in Pakistan, 2010 [cit. 2021-28-09]. Dostupné z: https://arxiv.org/ftp/arxiv/papers/1003/1003.1470.pdf.

[20] Salomon, D. Data Hiding in Text. In: *Data Privacy and Security*. Springer Science & Business Media, 2012, s. 465 [cit. 2022-23-03]. ISBN 978-1-4419-1816-1. Dostupné z: https://link-springer-com.ezproxy.lib.vutbr.cz/chapter/10.1007%2F978-0-387-21707-9_11.

[21] Sharma, S., Gupta, A., Trivedi, M. C. a Yadav, V. K. *Analysis of Different Text Steganography Techniques: A Survey* [online]. IEEE, 2016 [cit. 2021-6-10]. Dostupné z: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7546588.

[22] Sharma, S., Gupta, A., Trivedi, M. C. a Yadav, V. K. Analysis of Different Text Steganography Techniques: A Survey. In: *2016 Second International Conference on Computational Intelligence Communication Technology (CICT)*. 2016, s. 130–133

[cit. 2021-26-10]. DOI: 10.1109/CICT.2016.34. Dostupné z: https://ieeexplore.ieee.org/abstract/document/7546588.

[23] Thampi, S. M. *Information Hiding Techniques: A Tutorial Review*. 2008 [cit. 2021-26-10]. Dostupné z: https://arxiv.org/ftp/arxiv/papers/0802/0802.3746.pdf.

[24] Toftegaard, A. *A System for Hiding Steganography in Plain Sight* [online]. Kongens Lyngby, 2016 [cit. 2021-13-10]. Dostupné z: http://www2.imm.dtu.dk/pubdb/edoc/imm7049.pdf.

[25] Wai, e. a Khine, M. Modified Linguistic Steganography Approach by Using Syntax Bank and Digital Signature. *International Journal of Information and Education Technology*. Prosinec 2011, s. 410–415, [cit. 2021-26-10]. DOI: 10.7763/IJIET.2011.V1.68. Dostupné z: https://www.researchgate.net/publication/291409477_Modified_Linguistic_Steganography_Approach_by_Using_Syntax_Bank_and_Digital_Signature.

[26] Wai, e. a Khine, M. Syntactic Bank-based Linguistic Steganography Approach. *IPCSIT*. Říjen 2011, sv. 16, s. 108–113, [cit. 2021-26-10]. Dostupné z: https://www.researchgate.net/publication/267767675_Syntactic_Bank-based_Linguistic_Steganography_Approach.

[27] Wayner, P. *Disappearing cryphography: Information hiding: steganography and watermarking*. 2. vyd. Morgan Kaufmann Publishers, 2002 [cit. 2021-23-11]. 409 s. ISBN 1-55860-769-2.

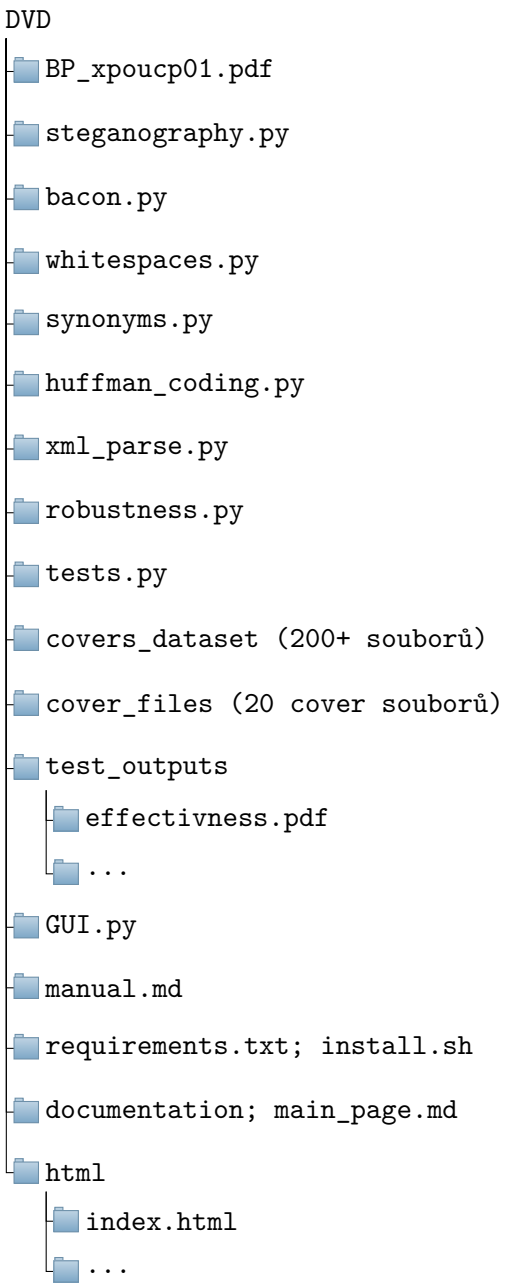
[28] Wikipedia, the free encyclopedia. *CardanGrille.png*. 2006 [cit. 2022-23-03]. Dostupné z: https://cs.wikipedia.org/wiki/Soubor:CardanGrille.png.

[29] Zielinska, E., Mazurczyk, W. a Szczypiorski, K. *Development Trends in Steganography* [online]. researchgate, 2012 [cit. 2021-11-10]. Dostupné z: https://www.researchgate.net/publication/258725207_Development_Trends_in_Steganography.

[30] ŽILKA, R. *Steganografie a stegoanalýza [online]*. 2008 [cit. 2021-10-21]. [cit. 2022-23-03]. 66 s. Diplomová práce. Masarykova univerzita, Fakulta informatiky, Brno. Vedoucí práce Říha, Z. Dostupné z: https://is.muni.cz/th/p3l0w/.

Příloha A

Struktura datového média



Soubor `BP_xpoucp01` v sobě obsahuje textovou část bakalářské práce. Zdrojový kód LaTeX, včetně všech ilustrací se nachází ve složce `text_src`.

Popis instalace a ovládání programu je popsán v textovém souboru `manual.md`. Instalace všech potřebných knihoven zajistí skript `install.sh`, který postupně vykoná příkazy v souboru `requirements.txt`.

Soubory s příponami `.py` jsou zdrojové soubory. Nachází se v nich implementace jednotlivých steganografických metod, testů a uživatelského rozhraní.

Ve složce `cover_dataset` se nachází všechny soubory, nad kterými jsem metody testoval. Pokud spustíme automatizované testy (soubor `tests.py`), vyhodnotí se všechny soubory ve složce `cover_files`. Do této složky jsem záměrně umístil menší množství souborů, jelikož testy jsou časově velmi náročné. Zejména co se týče testu robustnosti, tento test by pro větší množství vstupních souborů běžel několik desítek minut.

Ve složce `test_outputs` se nachází výstupy těhto testů. Kromě celkového výstupu je zde i například výstup šifrování stejného souboru všemi metodami (`same_file_max_message.txt`), na kterém můžeme pozorovat odlišnou efektivitu a kapacitu jednotlivých metod.

V souboru `documentation` a `main_page.md` jsou konfigurace pro nástroj `doxygen`[1], pomocí kterého generuji dokumentaci, která se nachází v souboru `index.html`.

[1] https://www.doxygen.nl/index.html