Методичні вказівки
до лабораторної роботи №3
«Комп'ютерна стеганографія»
з навчальної дисципліни вільного вибору
«Основи інформаційної та кібербезпеки»

Зміст практичного заняття: Вивчення методів комп'ютерної стеганографії для захисту інформації.

Загальні відомості

Стеганографія (це слово походить від грецьких слів *στεγανός (steganos)* (секрет, таємниця) і *γράφω (graphy)* (запис) і, таким чином, означає буквально "тайнопис") забезпечує обмін інформацією таким чином, що ховається сам факт існування секретному зв'язку. Вона не заміняє криптографію (шифрування даних), а доповнює її ще одним рівнем безпеки. При обробці даних стеганографічними методами відбувається приховування переданої інформації в інших об'єктах таким чином, що б стороння особа не здогадувалася про існування прихованого секретного повідомлення. При цьому, виявити таке повідомлення досить складно, але якщо це й відбудеться, то повідомлення може бути до того ж ще й надійно зашифроване.

Стеганографія являє собою сукупність методів та засобів їхньої реалізації, які базуються на різних принципах і дозволяють приховувати сам факт існування секретної інформації в тому або іншому середовищі. До неї можна віднести безліч секретних засобів зв'язку, таких як невидимі чорнила, мікрофотознімки, умовне розташування знаків, таємні (приховані) канали, засоби зв'язку з плаваючими частотами, голографія і т.д.

В даний час розвиваються методи комп'ютерної стеганографії самостійного наукового напрямку інформаційної безпеки, що вивчає проблеми створення компонент приховуваної інформації у відкритому інформаційному середовищі, яке може бути сформовано обчислювальними системами та мережами. Особливістю стеганографічного підходу є те, що він не передбачає прямого оголошення факту існування захищуваної інформації. Ця обставина дозволяє в рамках традиційно існуючих інформаційних потоків або інформаційного середовища вирішувати деякі важливі задачі захисту інформації ряду прикладних галузей.

Стеганографічні методи знаходять усе більше застосування в оборонній і комерційній сферах діяльності в силу їхньої легкої адаптації при вирішенні задач захисту інформації, а також відсутності явно виражених ознак засобів захисту, використання яких може бути обмежене або заборонене (як, наприклад, криптографічних засобів захисту).

Використання стеганографічних систем є найбільш ефективним при вирішенні проблеми захисту конфіденційної інформації. Так, наприклад, тільки одна секунда відцифрованого звуку з частотою дискретизації 44100 Гц і рівнем відліку 8 бітів у стереорежимі дозволяє приховати за рахунок заміни молодших розрядів на

приховуване повідомлення близько 10 Кбайт інформації. При цьому зміна значень відліків складає менше 1%. Така зміна практично не виявляється при прослуховуванні файлу більшістю людей. Приховування впроваджуваних даних, які у більшості випадків мають великий об'єм, висуває серйозні вимоги до контейнера: розмір контейнера в декілька разів повинен перевищувати розмір даних, що вбудовуються.

Крім прихованого передавання повідомлень, стеганографія є одним із найперспективніших напрямків для автентифікації і маркування авторської продукції ***з метою захисту авторських прав*** на цифрові об'єкти від піратського копіювання. На комп'ютерні графічні зображення, аудіо продукцію, літературні твори (програми в тому числі) наноситься спеціальна мітка, що залишається невидимою для очей, але розпізнається спеціальним програмним забезпеченням. Мітка містить приховану інформацію, що підтверджує авторство. Прихована інформація покликана забезпечити ***захист інтелектуальної власності***.

Нерідко методи стеганографії використовують для ***камуфлювання програмного забезпечення***. У тих випадках, коли використання програм незареєстрованими користувачами є небажаним, воно може буті закамуфльоване під стандартні універсальні програмні продукті (наприклад, текстові редактори) або приховане у мультимедійних файлах (наприклад, у звуковому супроводі комп'ютерних ігор).

1. Методи комп'ютерної стеганографії

Стеганографічна система або стегосистема – сукупність засобів і методів, які використовуються для формування прихованого каналу передачі інформації.

Контейнер – будь який файл, призначений для впровадження приховуваного повідомлення. Приховуване повідомлення – повідомлення, впроваджуване в контейнер. Повідомленням може бути секретний текст, зображення, фотографія, мітка, водяний знак. Стеганографічний канал (***стегоканал***) - канал передачі заповненого контейнера (Стего). ***Стегоключ*** (ключ) - секретні дані, використовувані в процесі впровадження приховуваного повідомлення в контейнері. На рисунку 1 представлена узагальнена модель стегосистеми і проілюстровані наведені визначення.

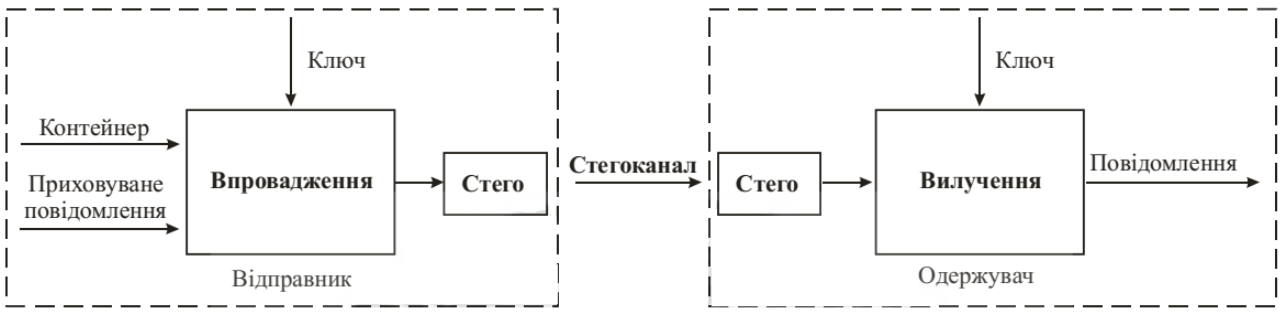


Рис. 1. Узагальнена модель стегосистеми

Методи комп'ютерної стеганографії розділяються на два класи:

- методи, засновані на надмірності інформації, (цифрова стеганографія);

- методи, засновані на використанні різних властивостей комп'ютерних форматів (форматна стеганографія).

До першого класу відносяться методи, які використовують молодші розряди цифрових відліків і методи, засновані на цифровій обробці сигналу. До другого класу відносяться методи видалення – ідентифікують заголовок файлу; методи впровадження інформації в невикористовувані області гнучких і жорстких дисків; методи впровадження інформації в текстові файли; методи, які використовують зарезервовані поля різних комп'ютерних форматів файлів.

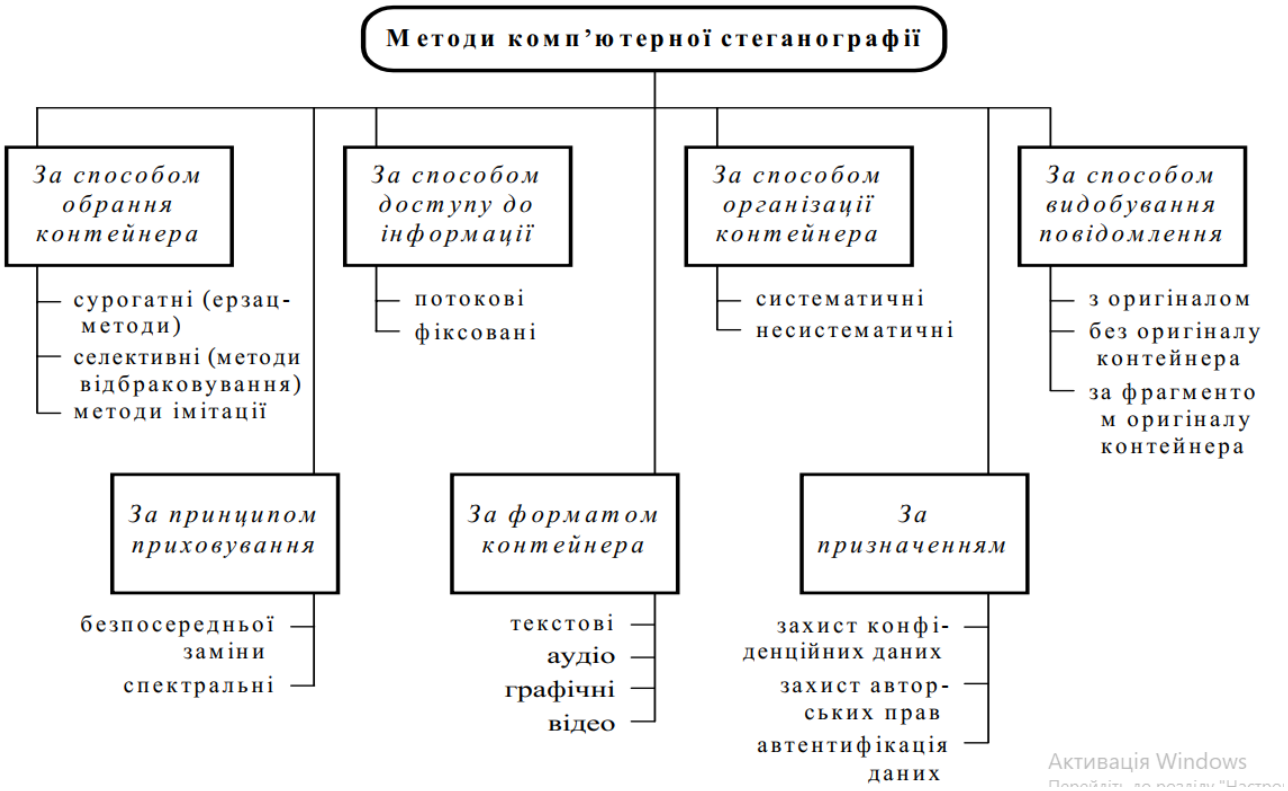


Рис.2. Класифікація методів комп'ютерної стеганографії.

За способом обрання контейнера розрізняють сурогатні, селективні та імітаційні методи стеганографії. В ***сурогатних*** (безальтернативних) методах стеганографії відсутня можливість вибору контейнера і для приховування повідомлення вибирається перший контейнер, що трапився, який у більшості випадків не є оптимальним для приховуваного повідомлення (так званий ерзац-контейнер). У ***селективних*** методах передбачається, що приховане повідомлення повинно відтворювати спеціальні статистичні характеристики шуму контейнера. Для цього генерують велику кількість альтернативних контейнерів, з наступним обранням (шляхом відбраковування) найоптимальнішого з них для конкретного повідомлення.

Окремим випадком такого підходу є обчислення деякої хеш-функції для кожного контейнера. При цьому для приховання повідомлення обирається той контейнер, хеш-функція якого збігається зі значенням хеш-функції повідомлення (тобто стеганограмою є обраний контейнер).

В ***імітаційних*** методах стеганографії контейнер генерується самою стеганосистемою. При цьому існують декілька варіантів реалізації. Так, наприклад,

шум контейнера може імітуватися приховуваним повідомленням. Це реалізується за допомогою процедур, які не лише кодують приховане повідомлення під шум, але й зберігають модель початкового шуму. У граничному випадку за моделлю шуму може будуватися ціле повідомлення.

За способом доступу до приховуваної інформації розрізняють методи ***потокових*** (безперервних) і ***фіксованих*** (обмеженої довжини) контейнерів.

За способом організації контейнери, подібно завадозахищеним кодам, можуть бути ***систематичними*** і ***несистематичними***. У перших можна вказати конкретні місця стеганограми, де знаходяться інформаційні біти власне контейнера, а де шумові біти, призначені для приховування інформації (як, наприклад, у широко поширеному методі найменшого значущого біту). У випадку несистематичної організації контейнера такий поділ не можливий. У цьому разі для виділення прихованої інформації необхідно обробляти вміст усієї стеганограми.

2. Типи контейнерів та стеганографічні алгоритми

Основне завдання будь-якої стегосистеми – вбудувати повідомлення в контейнер так, щоб сторонній спостерігач не міг помітити різницю між оригінальним та модифікованим контейнерами. Зазвичай система будується так щоб забезпечити певний компроміс її базових характеристик, до яких відносяться непомітність, стійкість, захищеність, пропускна здатність утвореного стеганоканалу та обчислювальна складність реалізації.

Робастність (стійкість)- це здатність прихованого повідомлення залишатися неушкодженим, навіть якщо стего-медіа піддають трансформації, лінійній та нелінійній фільтрації, масштабуванню, розмиванню, обрізанню та іншим атакам. Термін "робастність" (*robustness* - англ.) утворений від *robust* – міцний, грубий (англ.). Порівняйте з назвою одного із сортів кави – robusta. Мається на увазі, що робастні процедури повинні "витримувати" помилки, які тими чи іншими способами можуть потрапляти в вихідні дані або спотворювати передумови використовуваних ймовірнісно-статистичних моделей.

2.1. Графічні растрові зображення

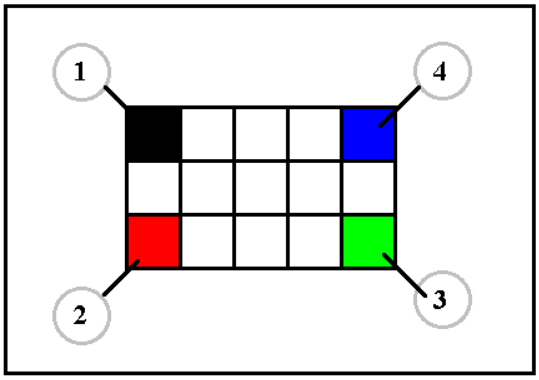
Метод дописування даних в BMP-файл.

Існує велика кількість способів прихованої передачі інформації в графічних файлах. Розглянемо можливість використання для цього особливості формату BMP. Структура файлу показана на малюнку. Два байти 42H і 4DH, представлені в шістнадцятковій системі числення, вказують на те, що формат даного файлу BMP. Відповідно до кодової таблиці CP-1251 ці числа після декодування дають латинські літери BM (тобто, графічний формат Bit Map).

Адреса	Заголовок															
00000000	42	4D	66	00	00	00	00	00	00	00	36	00	00	00	28	00
00000010	00	00	05	00	00	00	03	00	00	00	01	00	18	00	00	00
00000020	00	00	30	00	00	00	00	00	00	00	00	00	00	00	00	00
00000030	00	00	00	00	00	00	00	00	FF	FF	FF	FF	FF	FF	FF	FF
00000040	FF	FF	00	FF	00	00	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
00000050	FF	FF	FF	FF	FF	00	00	00	00	FF	FF	FF	FF	FF	FF	FF
00000060	FF	FF	FF	00	00	00										

Бітова карта

Шістнадцяткове число 66H, розташоване за адресою 02H, говорить про те, що розмір даного файлу дорівнює 102 байта. Це значення отримано шляхом переводу шістнадцяткового числа 66H в десяткову систему числення. Число 36H, записане за адресою 0AH, показує, з якої адреси починається запис картинки (це - зміщення від початку). За адресою 12H вказана ширина малюнка, виражена в пікселях. В даному випадку число пікселів рівне 5. Висота малюнка вказується в комірці 16H (для обраного малюнка 3 пікселі). В комірці 1AH вказано число площин на малюнку - 1. За адресою 1CH вказана глибина кольору. В даному випадку число 18H говорить про те, що для формування колірних відтінків цього малюнка використовується 24 біта. В комірці 22H вказується об'єм пам'яті, необхідний для запам'ятовування бітової карти (об'єм малюнка без службової інформації).



Наведений файл містить малюнок, показаний зліва.

Малюнок складається з 15-ти пікселів (прямокутник 5x3). Одинадцять пікселів білі, піксель в лівому верхньому кутку прямокутника чорний (1), в лівому нижньому кутку червоний (2), в правому нижньому - зелений (3) і у верхньому правому - синій (4).

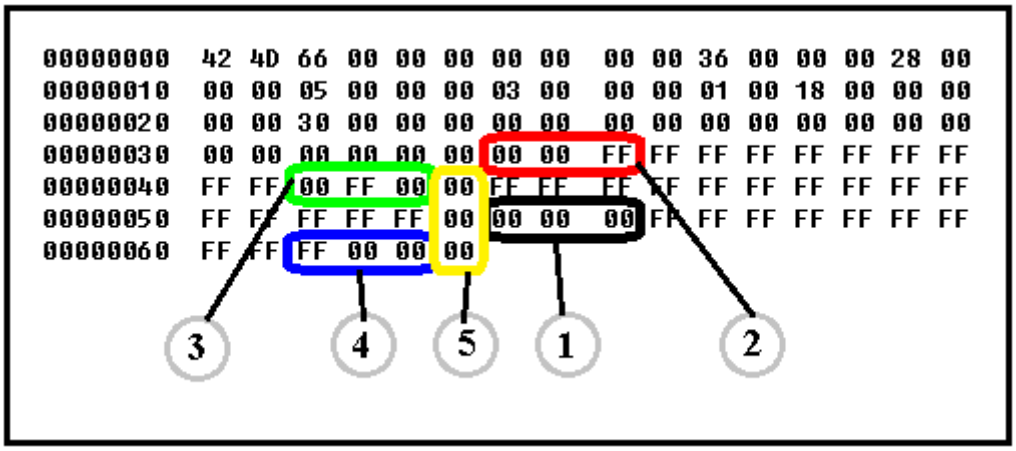
Запис бітової карти в пам'ять починається з лівого нижнього кута малюнка. Червоний піксель (2) описується складовими R = 255, G = 0, B = 0. Запис в пам'ять (при збільшенні адреси комірки) ведеться в зворотному порядку B-G-R. Таким чином, в найпершу комірку бітової карти (адреса 36) заноситься синя складова B = 00. У комірку 37 записується зелена складова червоного пікселя G = 00, а в клітинку 38 червона складова R = FF (див. Наступний малюнок). На малюнку червоний піксель позначений цифрою 2.

Наступні три пікселя в нижньому рядку білі. Тому чергові дев'ять байт мають максимальне значення FFH (255D). В комірках 42, 43 і 44 розміщуються три байта зеленого пікселя (цифра 3).

В комірці 45 розміщений байт 00 - це доповнення, призначене для вирівнювання рядків дампу пам'яті. Ця комірка надлишкова, вона не несе ніякої корисної інформації, але її вміст передається разом з малюнком.

Наступні п'ять пікселів малюнка (другий рядок) - білі. Ці пікселі описуються за допомогою п'ятнадцяти байтів FFH, які розміщені в комірках пам'яті 46H ... 54H. В комірці 55H міститься вирівнюючий байт 00.

В комірках 56, 57 і 58 розміщуються байти 00 - це колірні компоненти чорного пікселя. Далі на малюнку розміщені 3 білих пікселі верхнього рядка. Їм відповідають дев'ять байт FFH.



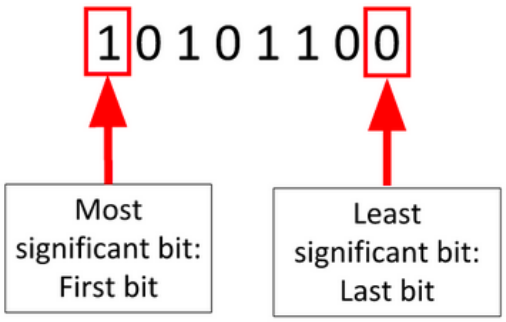
В комірках 62, 63 і 64 розміщуються колірні складові синього пікселя.

Комірка 65 використовується для вирівнювання. Три додаткових байти позначені цифрою 5.

Додаткові комірки з'являються в тих випадках, коли добуток кількості пікселів на три не кратний числу 4. Іншими словами, довжина даних, що описують окремий рядок малюнка, повинна бути кратна чотирьом.

Саме додаткові байти в графічному файлі, призначені для вирівнювання рядків, можуть бути використані для прихованої передачі інформації в графічному файлі.

LSB (Least Significant Bit, найменший значущий біт) – суть цього методу полягає в заміні останніх значущих бітів у контейнері (зображення, аудіо або відеозапису) на біти приховуваного повідомлення. Різниця між порожнім і заповненим контейнерами повинна бути не відчутна для органів сприйняття людини.



Для простоти пояснення розглянемо графічний контейнер – зображення у форматі BMP. В даному форматі для опису кожної точки (пікселя) використовуються 3 байти, що позначають в якій пропорції необхідно змішувати червоний, зелений і блакитний кольори (колірна схема RGB). Якщо зробити заміну старших біт в цих байтах, колірні зміни в зображенні будуть кидатися в очі. Молодші ж біти дають куди більш незначний вклад в зображення. Якщо використовувати по одному молодшому біту в кожному кольорі для запису приховуваного повідомлення, то розпізнати зміни людське око буде не здатне.

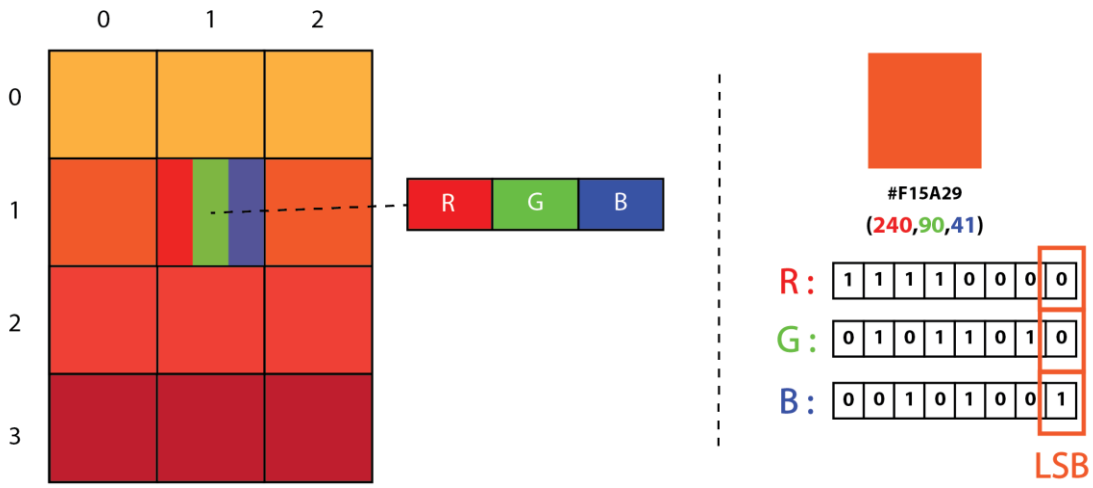


Рис. 3. Представлення зображення як двовимірного масиву пікселів RGB.

Таким чином, в графічному зображенні розміром 117 Кбайт (200 на 200 пікселів) можна сховати, як мінімум, 14 Кбайт прихованої інформації.

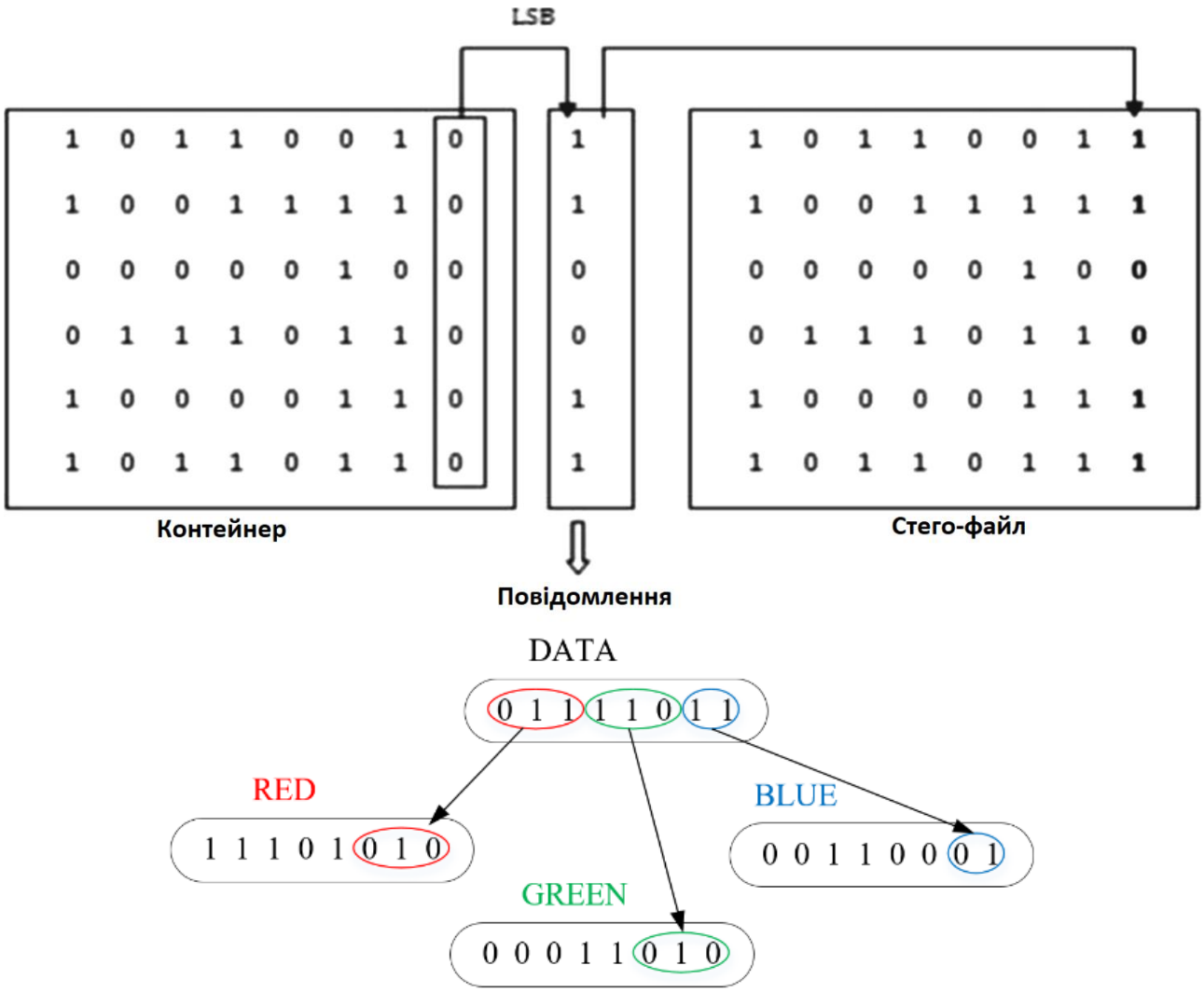


Рис. 4. Приклади кодування LSB.

Була описана найпростіша варіація методу LSB, яка має безліч недоліків. Приховане повідомлення легко зруйнувати, стискаючи або відображаючи зображення. При такому підході не забезпечується таємність вбудовування повідомлення: точно відомо місце розташування інформаційних бітів (кожен крайній з кінця біт).

Недоліком методу LSB є також чутливість до розміру зображення, тобто чим менший розмір зображення, тим більше будуть відрізнятися два сусідні пікселі, тому пропонується використовувати зображення з великою роздільністю. Також метод «видає себе» при побітовому перегляді зображення, де чітко видно області зображення, в які «вбудовано» таємну інформацію. Попри це, метод запису Least Sagnificant Bit є досить популярним, стійким та простим під час реалізації.

Стеганографічне зображення, отримане в результаті роботи LSB-алгоритму, дуже чутливе до будь-яких модифікацій (тобто низький рівень робастності). Найменше опрацювання цього зображення призведе до втрати вбудованої інформації.

Підвиди LSB-алгоритмів для растрових зображень.

BlindHide (приховування наосліп). Найпростіший алгоритм: дані записують, починаючи з верхнього лівого кута зображення до правого нижнього — піксель за пікселем. Приховані дані програма записує у бітах кольорів пікселя. Приховані дані розподіляються у контейнері нерівномірно. Якщо приховані дані не заповнять повністю контейнер, то лише верхня частина зображення буде засміченою.

HideSeek (заховати-знайти). Цей алгоритм у псевдовипадковий спосіб розподіляє приховане повідомлення у контейнері. Для генерації випадкової послідовності використовує пароль. Дещо «розумніший» алгоритм, але все ж не враховує особливостей зображення-контейнера.

FilterFirst (попередня фільтрація). Виконує фільтрацію зображення-контейнера — пошук пікселів, у які записуватиметься прихована інформація (для яких зміна розрядів буде найменш помітною для ока людини).

BattleSteg (стеганографія морської битви). Найскладніший і найдосконаліший алгоритм. Спочатку виконує фільтрацію зображення-контейнера, після чого прихована інформація записується у «найкращі місця» контейнера у псевдовипадковий спосіб (подібно, як у HideSeek).

Інші методи приховування інформації в графічних файлах орієнтовані на формати файлів з втратою, наприклад, JPEG. На відміну від LSB вони більш стійкі до геометричних перетворень. Це виходить за рахунок варіювання в широкому діапазоні якості зображення, що призводить до неможливості визначення джерела зображення.

2.2. Аудіофайли

LSB. Принцип роботи цього методу для аудіофайлів аналогічний, як для растрових зображень. У разі аудіофайлів для запису інформації використовуються молодші біти кожного звукового семплу.

Polarity Inversion (інверсія полярності). У більшості мов потік повітря, що утворюється під час звукоутворення, є однонапрямленим. Це викликає постійну полярність коливань мовлення. Людська слухова система не в змозі розрізнити мовні сигнали з додатною й від'ємною полярністю. Цей факт використовується у цьому методі.

Аудіосигнал ділиться на сегменти (зазвичай для мовленнєвого сигналу це склади) і кожному сегменту призначається один біт інформації. Значення біта задається зміною полярності сегмента.

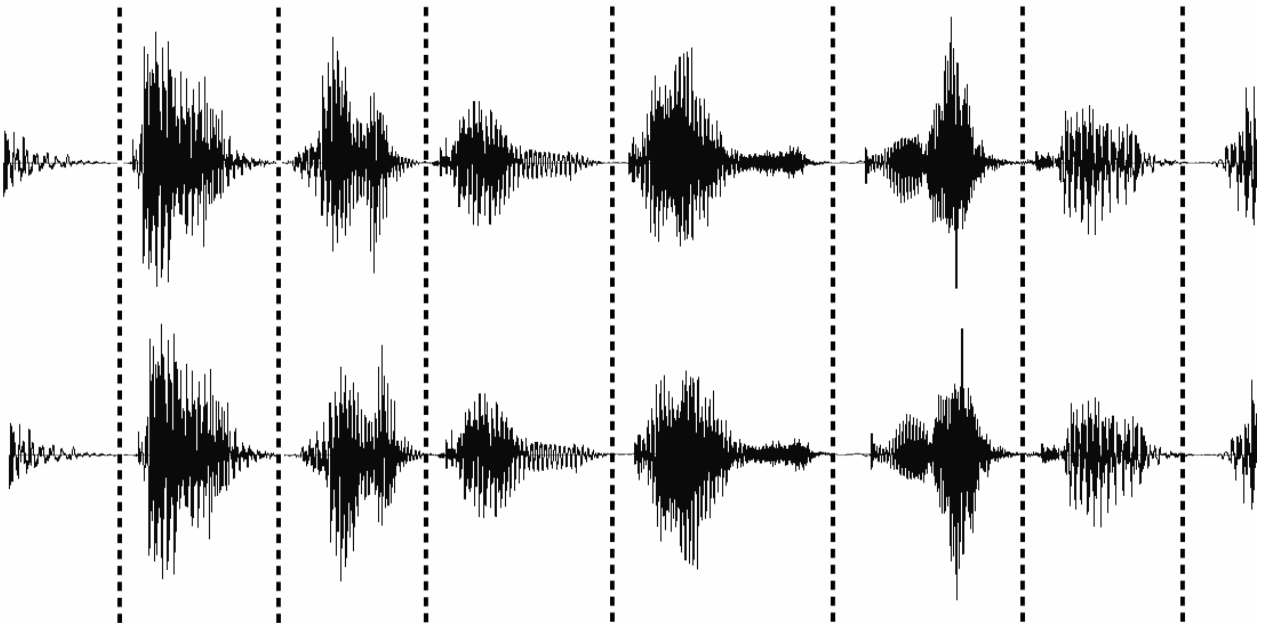


Рис. 5. Аудіосигнал до (зверху) та після (знизу) використання алгоритму інверсії полярності. Пунктиром розділено сегменти аудіосигналу

Echo Hiding (ехо-приховування). Цей метод виконує приховування даних додаванням відлуння у звуковий сигнал. До параметрів відлуння належать: початкова амплітуда, часу спаду і зсуву (час затримки між вихідним сигналом і його відлунням). При зменшенні зсуву два сигнали змішуються і відлуння перестає бути відчутним для людини, зазвичай, якщо відстань між ними близько 1 мс.

Аудіосигнал розділяють на сегменти, кожному з яких призначається біт. Далі для кожного сегмента виконується зміна параметрів відлуння. Використовують два часи затримки: один – для кодування нуля, другий – для кодування одиниці

Phase Coding (фазове кодування) – вбудовування інформації в фазу сигналу. У цьому методі фаза початкового сегмента аудіосигналу модифікується залежно від вхідних даних. Фази наступних сегментів узгоджуються з нею для збереження різниці фаз. Це необхідно, тому що людське вухо дуже чутливе до різниці фаз.

Фазове кодування є одним з найефективніших способів кодування за критерієм відношення сигнал–шум. Головним недоліком можна вважати низьку пропускну здатність.

Spread Spectrum (SS, розширення спектра). За методом розширення спектра намагаються якомога більше «рознести» приховану інформацію по частотному спектру сигналу звуку. Це аналогічно до систем, що використовують реалізацію LSB, яка у випадковий спосіб поширює біти вхідних даних в усьому спектрі. Проте, на

відміну від кодування LSB, метод SS розміщує секретне повідомлення над частотним спектром звукового файлу, використовуючи код, який не залежить від звукового сигналу. В результаті кінцевий сигнал займає більшу смугу пропускання, ніж насправді необхідно для передавання [13].

Tone Insertion (вставка тонів). Метод вставки тонів ґрунтується на поганій чутності та нечіткості низьких тонів у присутності компонент значно вищого спектра.

2.3. Відеофайли

Дискретно-косинусне перетворення (ДКП). Цей метод виконує зміну коефіцієнтів дискретного косинусного перетворення. Дискретно-косинусне перетворення використовується для алгоритму стиснення JPEG для перетворення блоків пікселів зображення 8 x 8 на 64 ДКП коефіцієнти.

Один зі стегоалгоритмів використовує таку особливість. Стиснені дані зберігаються як цілі числа, проте усі обчислення виконуються з числами з плаваючою комою, після чого їх округляють. Помилки округлення визначають ступінь втрати при стисненні. Цей алгоритм використовує рівні округлення для приховування інформації.

2.4. Неформатований текст

Маніпулювання пробілами між словами. Різна кількість пробілів між словами може приховувати інформацію. Наприклад, нулю відповідає один пробіл, одиниці – два.

Додавання хвостових пробілів. У кінець кожного речення чи абзацу додають певну кількість пробілів. Наприклад, нуль пробілів у кінці речення відповідає бітовій послідовності 000, сім пробілів – 111.

Заміна символами з іншої абетки. Цей метод передбачає заміну кириличного символу на символ латиниці з таким самим виглядом (значення біта – нуль) або відмову від такої заміни (значення біта – одиниця).

Морфологічно-синтаксичний метод. Послідовність речень розглядають як послідовність бітів. Функція, яка визначає, чи речення відповідає нулю, чи одиниці, може бути найрізноманітнішою.

Наприклад, чи речення активне, чи пасивне; чи речення містить два іменники; чи останній біт хешу речення є нулем, чи одиницею тощо. Складний для реалізації метод, через специфіку вбудовування прихованої інформації: на основі прихованої інформації формують контейнер (можливо, переформульовуючи речення контейнера, зберігаючи при цьому їхню семантику). Особливою відмінністю методу є те, що прихована інформація зберігається у разі переведення контейнера з одного формату в інший, навіть при друці на папір.

2.5. Форматований текст

Форматування символів тексту розглядають як бітову послідовність. Форматування, яке відповідає нулю, візуально дуже близьке до форматування, що відповідає одиниці (різниця непомітна для неозброєного ока). Наприклад, нулю

відповідає розмір символу в 14 кеглів, одиниці – 14,5 кегля, або нулю – колір RGB(0, 0, 0), одиниці – RGB(0, 0, 1).

2.6. Стиснені (заархівовані) файли

Файли з секретними даними стискають і додають в кінець файлу-архіву так, що ці файли є “невидимими” для звичайних програм-архіваторів. Цей метод використовує нестандартний алгоритм створення скомпресованих файлів і його (з відповідними модифікаціями) також можна застосовувати для файлів-контейнерів інших типів.

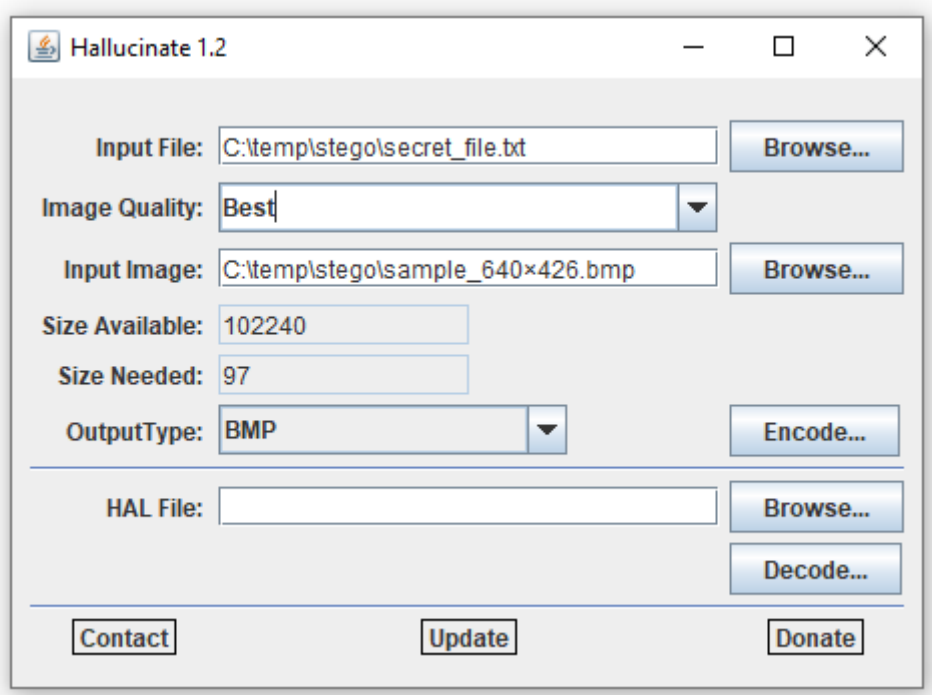
3. Робота із стеганографічними програмами

Для впровадження цифрових даних в мультимедійні контейнери розроблено велику кількість комерційних і любительських програм. Наприклад, S-Tools, JPHS, Open Puff, MP3Stego, StegoMagic та ін. (https://www.jjtc.com/Steganography/tools.html).

3.1. Hallucinate

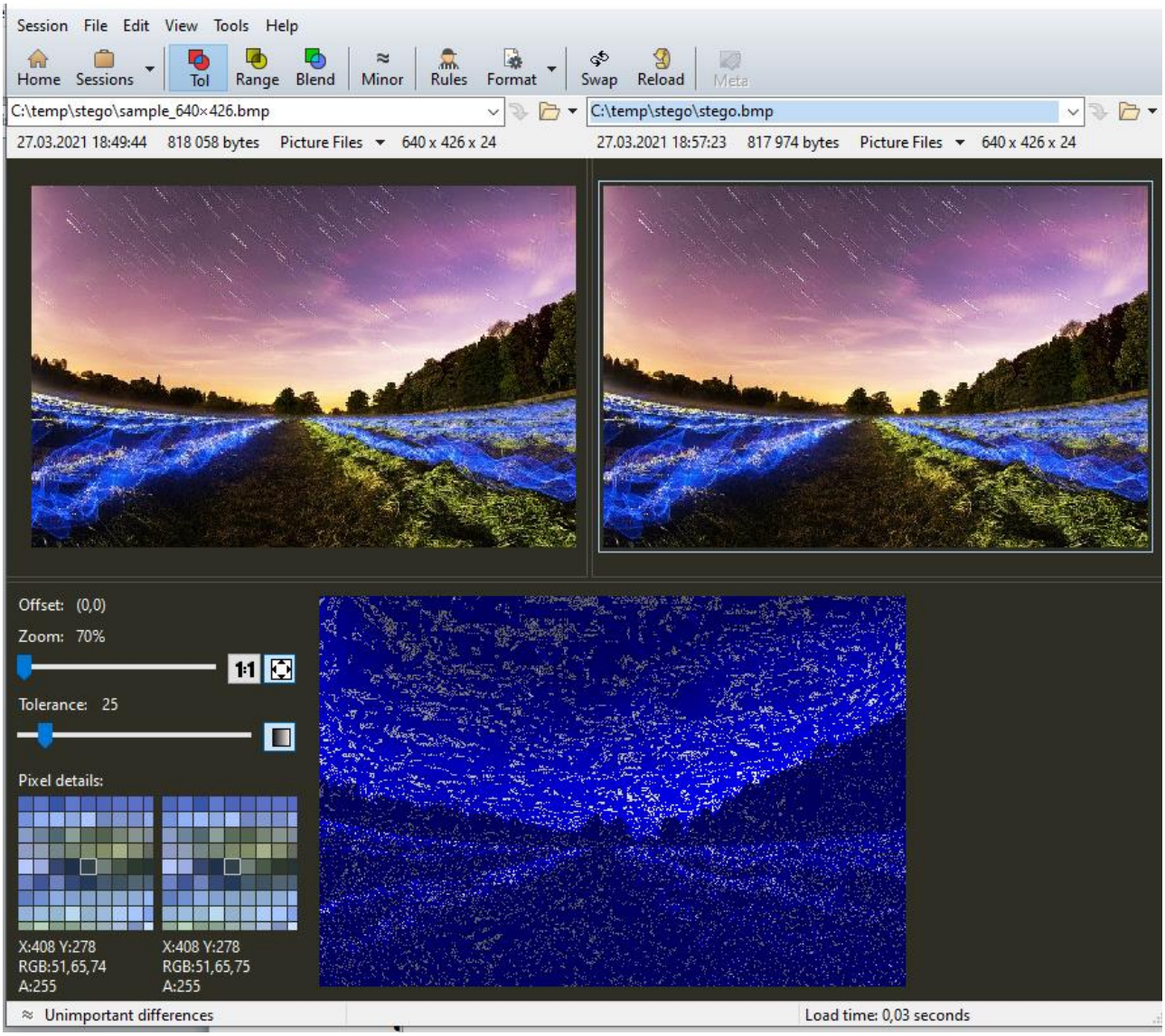
Остання версія програми Hallucinate (v. 1.2) вийшла в листопаді 2015 року (https://sourceforge.net/projects/hallucinate/). Ця компактна (всього 34 Кбайт) утиліта написана на Java і не вимагає установки. В якості контейнера вона підтримує формати BMP і PNG. Зображення у форматі PNG використовуються сьогодні куди частіше, ніж BMP. Їх багато навіть у тимчасових каталогах браузера, тому лежати самотнім і дуже помітним файлом на диску такий контейнер точно не буде.

Інтерфейс у Hallucinate простий і функціональний. Потрібно вибрати контейнер (**Input Image**), вказати приховуваний в ньому файл (**Input File**) і бажаний ступінь якості підсумкової картинки (**Input Quality**). Доступні вісім варіантів. Чим сильніше обробляється вихідне зображення, тим більше в ньому можна сховати, але тим помітнішими стають артефакти. Виберемо в налаштуваннях кращу якість і проілюструємо цю різницю, виконавши операцію з файлом BMP.



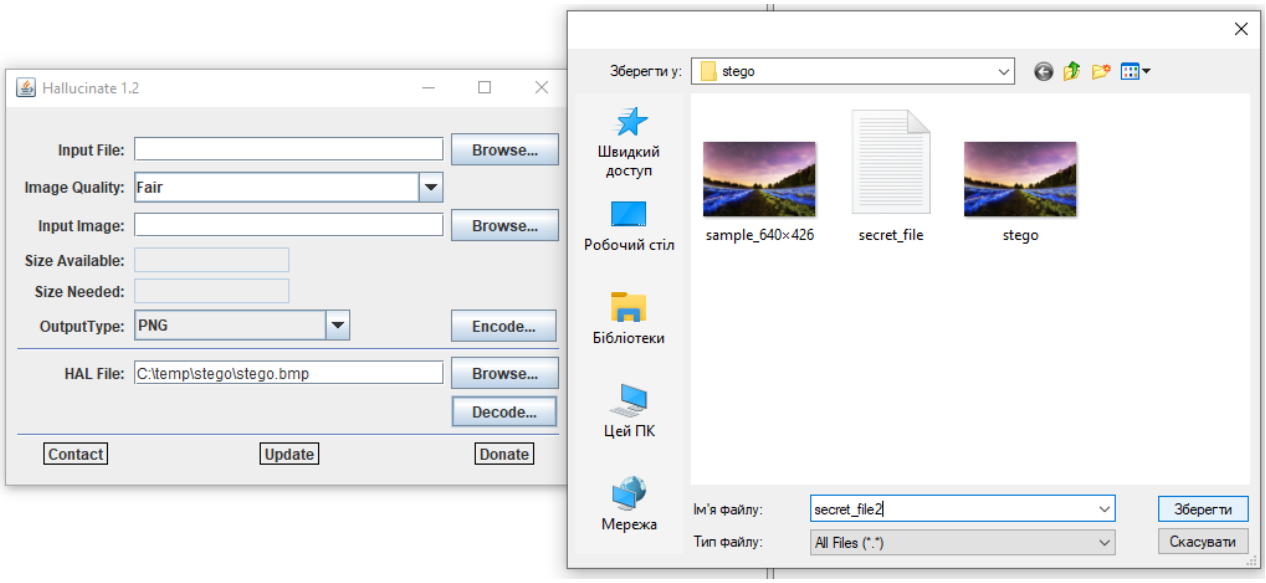
Для створення стего-файлу натискаємо кнопку **Encode…**, вибираємо ім’я файлу та **OK**.

Візуально картинки ліворуч і праворуч не відрізняються. Однак програма **Beyond Compare** (https://www.scootersoftware.com/download.php) показує різницю між ними.



Текстовий файл закодований як зміни яскравості окремих пікселів, рівномірно розподілених по всьому кадру. Тільки в найтемніших і найсвітліших ділянках вони гуртуються щільно.

Витягується прихований файл дуже просто. Досить вибрати контейнер (**HAL-file** в термінології автора програми), натиснути **Decode** і вказати місце для збереження файлу.

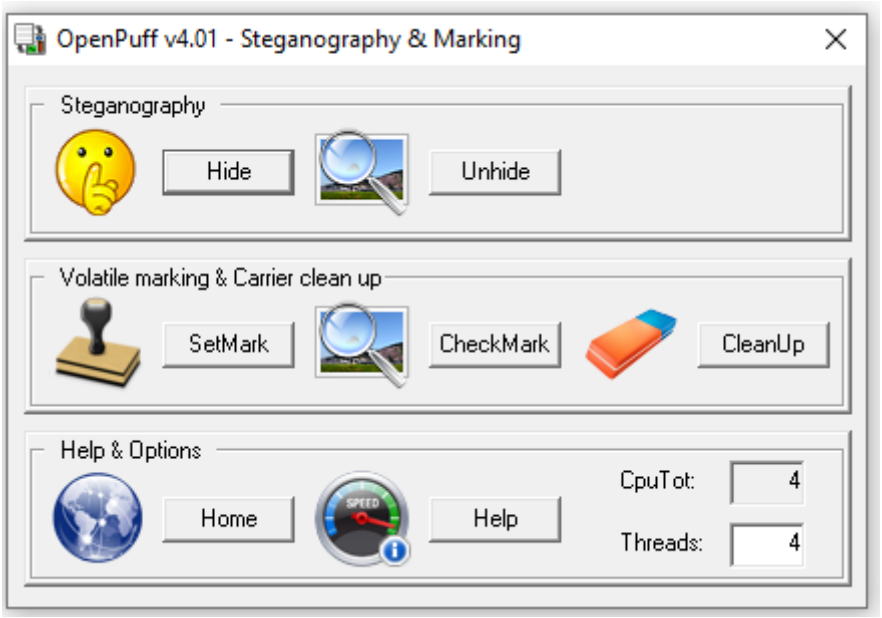


3.2. OpenPuff

Найкращою з погляду практичності є програма **OpenPuff** (https://embeddedsw.net/OpenPuff_Steganography_Home_ua.html). Вона підтримує найбільше серед інших програм форматів даних (зокрема скомпресовані формати растрових зображень та аудіо). Програма може використовувати складений контейнер, що дає змогу приховувати великі обсяги даних.

Можна приховувати як будь-які файли, так і папки, причому декілька водночас. Приховані дані програма архівує та шифрує. Програма має високу швидкодію навіть за великих обсягів даних.

Сторінка завантаження: https://embeddedsw.net/OpenPuff_download.html. Можна зробити невелику пожертву і підтримати проект. Запускаємо програму.



За запитом користувача OpenPuff розділить важливий файл на фрагменти, виконає ретельне шифрування даних і організовує їх зберігання у файлах-контейнерах. Тепер сторонній користувач, який вирішив вивчити вміст диска або

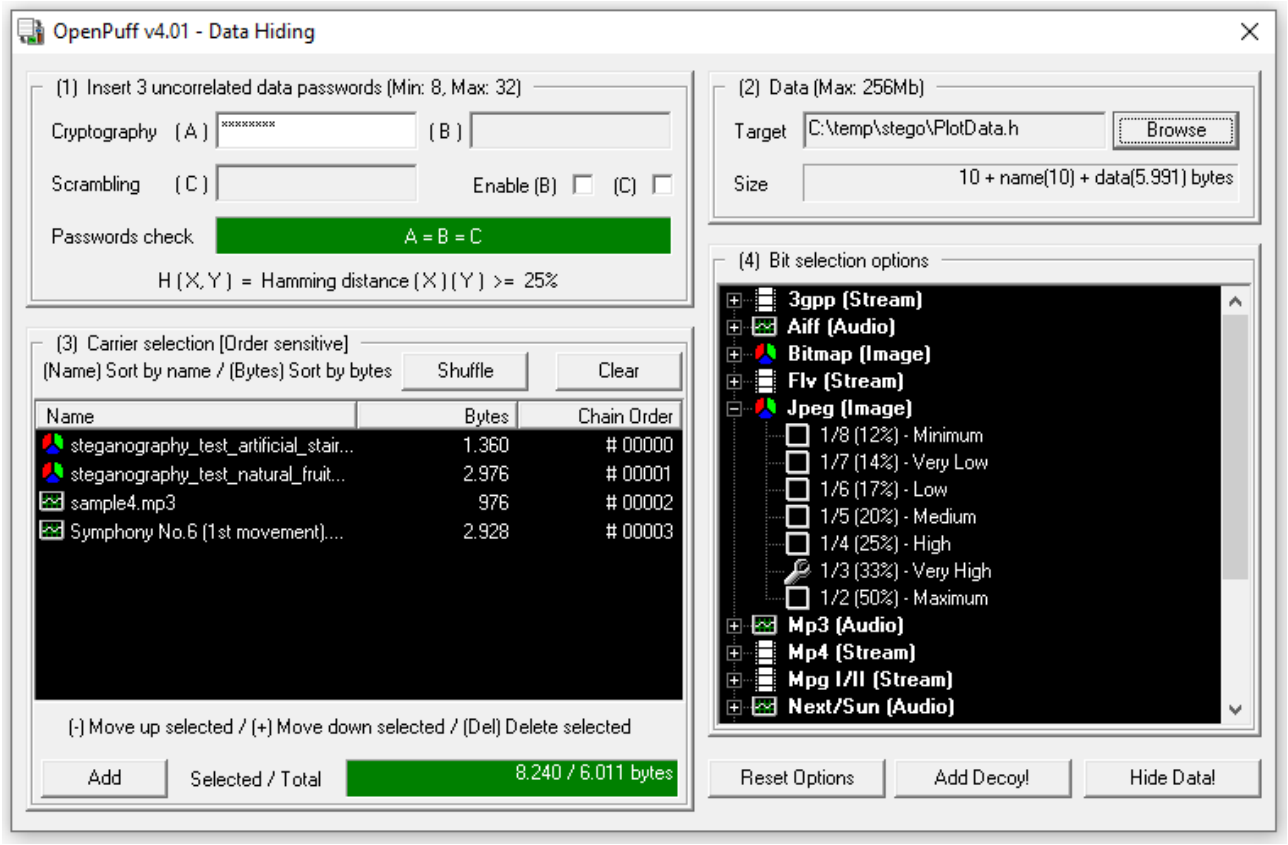
портативного носія, не побачить нічого крім звичайних відео, аудіо та іншого необразливого контенту. Програма підтримує широкий спектр поширених форматів (включаючи BMP, JPG, PNG, MP3, WAV, MP4, MPG, FLV, SWF, PDF та багато інших).

Натискаємо кнопку **Hide**.

На відміну від інших утиліт, що підтримують парольний захист приховуваного повідомлення, OpenPuff вміє використовувати для шифрування криптографічно стійкий генератор псевдовипадкових чисел (CSPRNG – Cryptographically secure pseudorandom number generator). Якщо простого пароля недостатньо, то постав прапорці навпроти полів B і C, а потім введи в них три різних пароля довжиною від 8 до 32 символів. На їх основі CSPRNG згенерує унікальний ключ, яким і буде зашифровано повідомлення.

У лівому верхньому кутку, потрібно заповнити три паролі. При бажанні другий і третій можна відключити, знявши позначки з **Enable** (B) і (C). У другій секції, вам потрібно вибрати файл (**Target**), який ви хочете заховати.

У третій секції, вам потрібно вибрати файл (низку файлів), який буде контейнером для зберігання ваших даних (кнопка **Add**). Дрібні файли можна зберігати в картинках і аудіозаписах, а великі зручніше ховати в відеозаписах – OpenPuff підтримує MP4, MPG, VOB і безліч інших форматів. Максимальний розмір приховуваного файлу - 256 Мбайт.

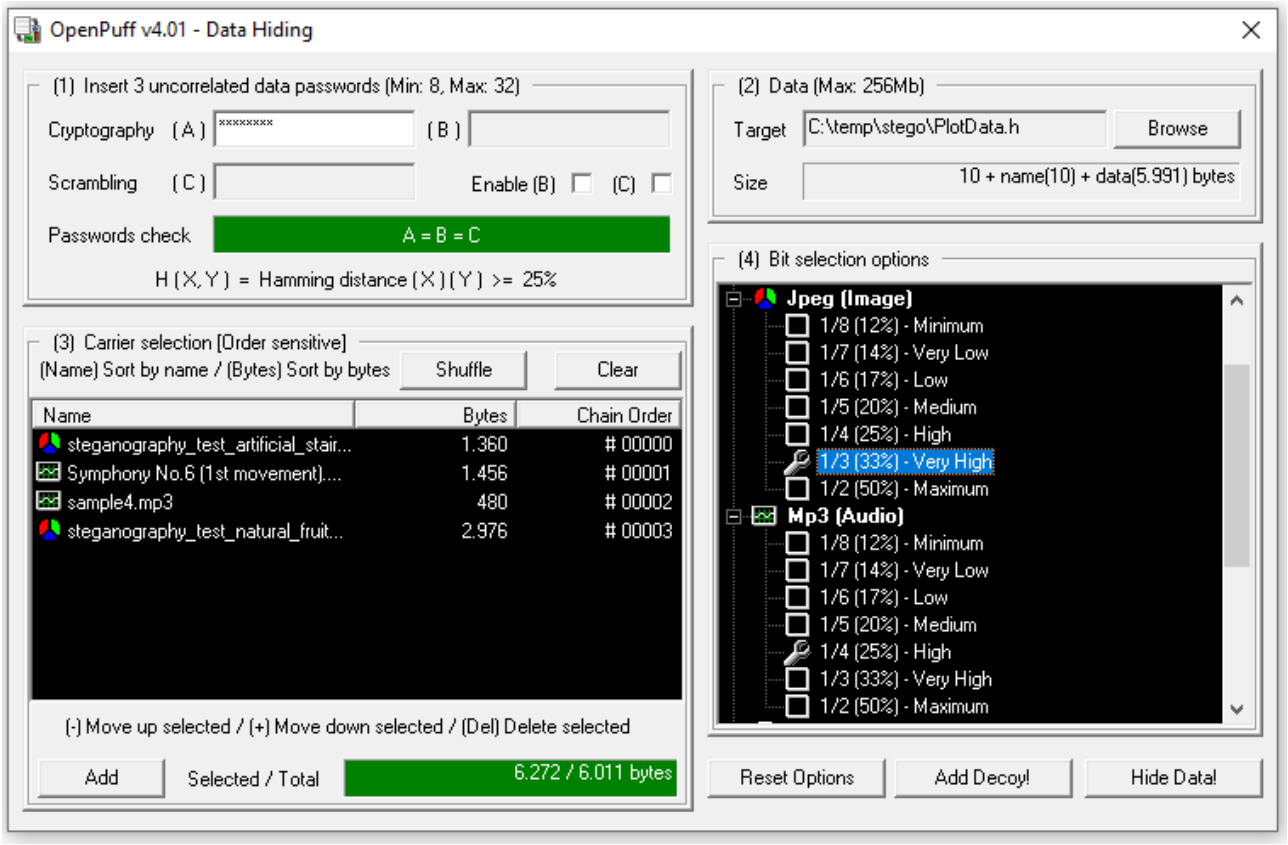


Ви повинні переконатися, що розмір контейнера, перевищує розмір файлу, який ви хочете приховати. Для цього ви можете внести зміни в секції вибору біт, або збільшити кількість файлів-контейнерів. У правій частині вікна вбудовування можемо також задати ступінь впровадження (заміни оригінальних пікселів на вбудовані з

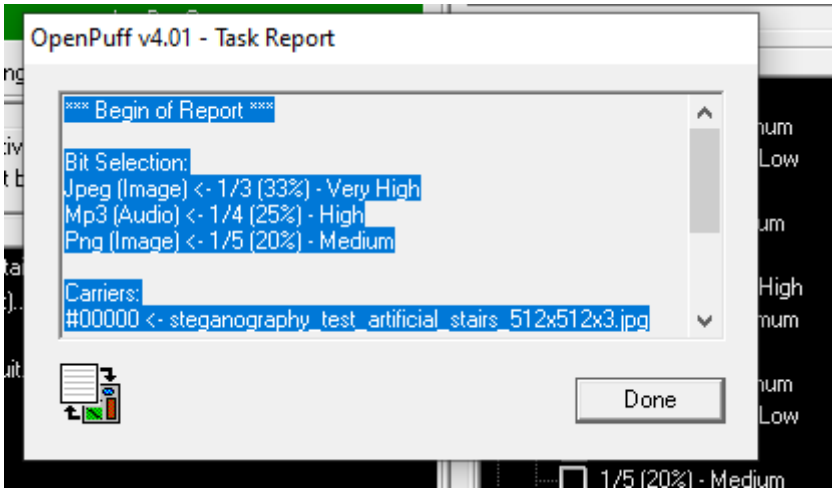
файлу-повідомлення). Задається вона від мінімуму (12% – замінюється тільки один незначущий біт з восьми) і максимум (50%), за замовчуванням стоїть значення 20%.

*Цікава опція – додати приманку (**Add Decoy**!), По суті: ще один контейнер і пароль, які ви могли б розкрити, якщо хтось помітив приховані дані у файлі, таким чином зберігши «друге дно» в таємниці.*

Натиснути на кнопку **Hide data** (Приховати дані) у вікні 4 і вказати директорію збереження кінцевого файлу.

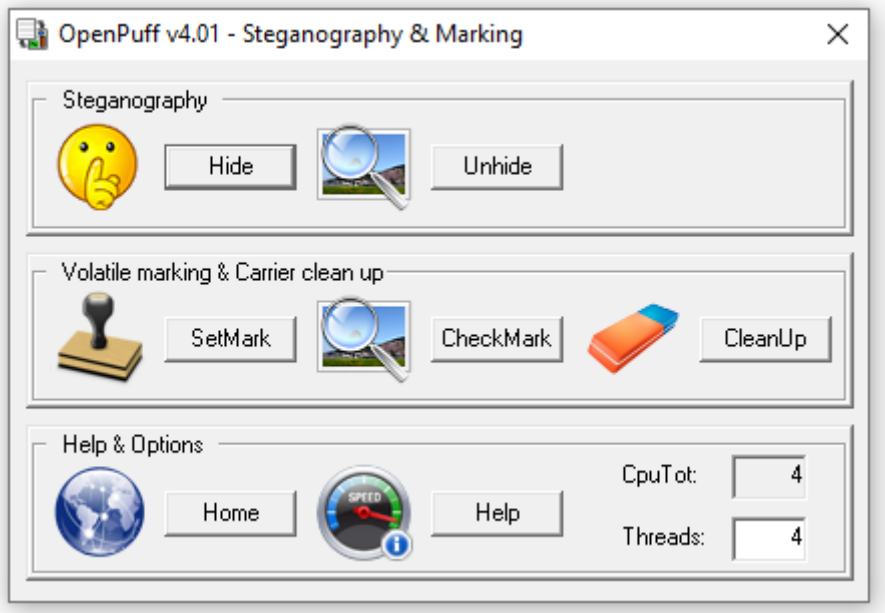


Велике значення має порядок ланцюжка стеганоконтейнерів. При видобуванні файлу-повідомлення він має бути незмінним, як і паролі. Вся ця інформація зберігається у протоколі завершення завдання, як і ступінь заміни.



Видобування файлу-повідомлення із ланцюжка стеганоконтейнерів відбувається у такій же послідовності. Як і в процесі вбудовування файлів. Попередньо створюється папка, куди має бути проведене видобування файлу-повідомлення. Протокол завершення показаний на наступному рисунку.

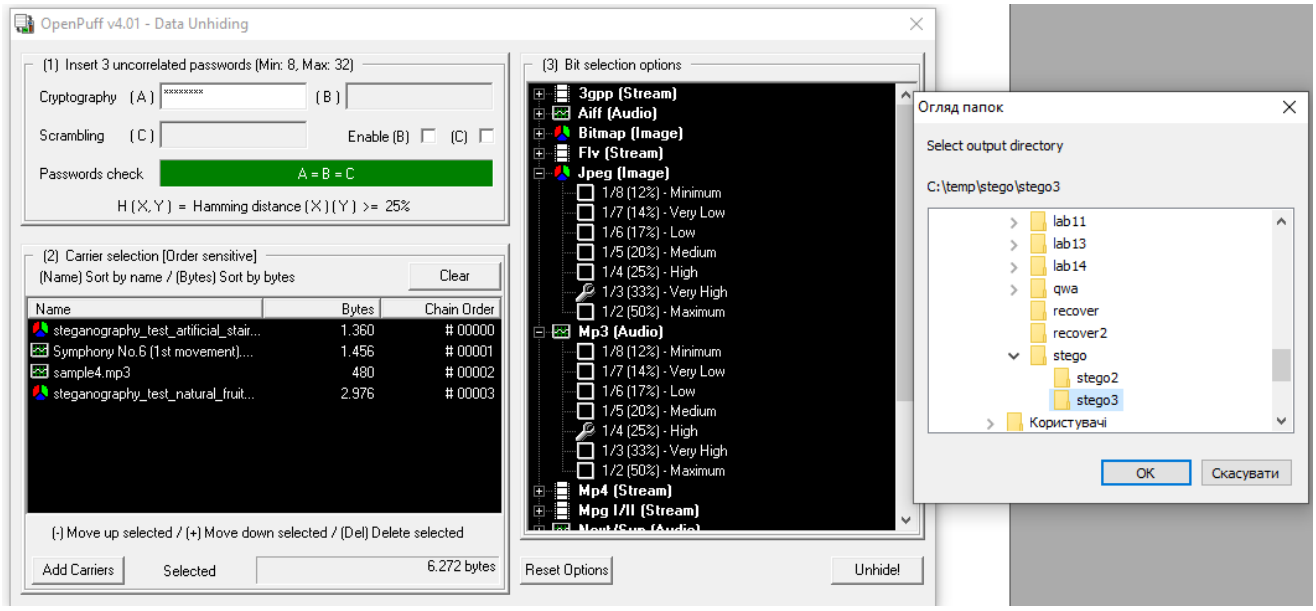
У початковому меню програми натиснути на кнопку **Unhide** (Розкрити).



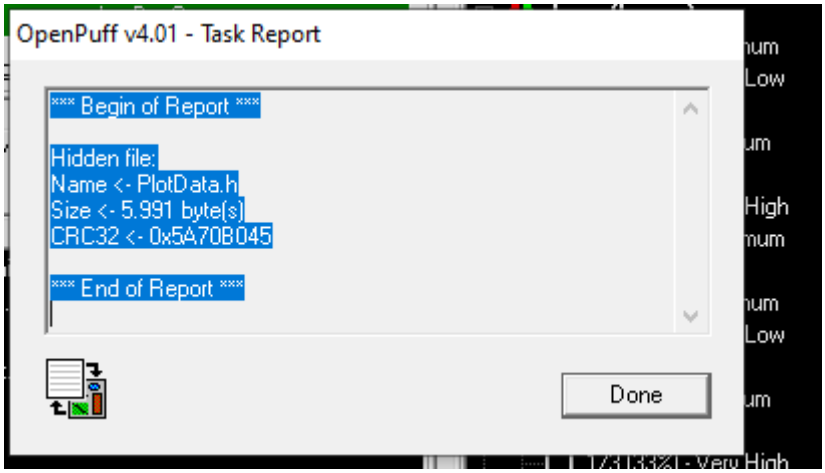
У вікні програми знаходиться кілька пронумерованих вікон. У вікні номер 1 ввести три зазначених в завданні паролі.

У вікні номер 2 вибрати контейнери в правильному порядку.

У вікні номер 3 відповідно встановити Bit selection options (Налаштування вибору бітів) для контейнерів. Натиснути на кнопку **Unhide** (Розкрити) і вказати шлях для збереження вилученого файлу.



Протокол завершення завдання показаний на наступному рисунку.



Ви можете використовувати OpenPuff для створення і читання стегоміток, своєрідних «водяних знаків», тільки для цифрової інформації. Виберіть в головному меню **SetMark**.

У першій секції додайте текст, а в другій додайте файли, до яких його потрібно прикріпити, тиснете **SetMark**!

Тепер у разі виникнення будь-яких правових спорів, досить завантажити файл в програму і натиснути **CheckMark**. На жаль, конвертація або зміна формату файлу, ведуть до стирання стегоміток.

Завдання до лабораторної роботи

Завдання А. Робота із стеганографічними програмами.

Файли-контейнери для виконання завдань можна завантажити із сайтів зі зразками файлів, наприклад: https://www.learningcontainer.com/, https://file-examples.com/, https://filesamples.com/ ...

А1. Впровадження текстового файлу в графічний контейнер.

1. Використовуючи програму **Hallucinate**, приховати в графічному контейнері (BMP файл) текстовий файл.
2. Використовуючи програму Hallucinate, витягти текстовий файл з графічного контейнера.
3. У звіті опишіть порядок виконання завдання.

А2. Впровадження рисунка в графічний контейнер.

1. Використовуючи програму Hallucinate, приховати в графічному контейнері (BMP файл) інший графічний файл.
2. Використовуючи програму Hallucinate, витягти графічний файл з графічного контейнера.
3. Вбудуйте в контейнер файл трохи менше допустимого розміру.
4. Опишіть результат. Вилучіть файл із контейнера.
5. У звіті опишіть порядок виконання завдання.

А3. Вивчення алгоритму приховування, який використовується в програмі Hallucinate (детальний порядок виконання цього завдання у Додатку 1).

1. У графічному редакторі MS Paint створити малюнок (прямокутник або квадрат) із заданими атрибутами і вказаним кольором заливки паперу (див. Табл. А1). Для прямокутника спочатку вказана його ширина, а потім - висота. Наприклад, розмір 10x4 означає, що прямокутник має ширину: 10 точок (пікселів) і висоту 4 точки.
2. Впровадити в малюнок (помістити в контейнер) за допомогою програми Hallucinate заданий текст.
3. У звіті представити бітові карти малюнка без тексту і з текстом. Бітові карти слід отримати за допомогою програми Hex Editor Neo 6.11 (https://www.hhdsoftware.com/free-hex-editor) або онлайн https://hexed.it/.

Номер варіанту визначається останньою цифрою Вашої залікової книжки.

Таблиця А1.

№ варіанту	Малюнок	Текст
0.	Чорний прямокутник 10x4 пікселів	fML
1.	Білий прямокутник 10x4 пікселів	haL
2.	Чорний прямокутник 10x4 пікселів	WAu
3.	Білий квадрат 10x10 пікселів	AiS
4.	Чорний квадрат 10x10 пікселів	dWq
5.	Білий прямокутник 10x4 пікселів	Asd
6.	Чорний прямокутник 10x4 пікселів	MiL
7.	Білий квадрат 10x10 пікселів	WtS
8.	Чорний квадрат 10x10 пікселів	CiA
9.	Білий прямокутник 10x4 пікселів	Saw

А4. Впровадження текстового файлу в послідовність контейнерів.

1. Використовуючи програму **OpenPuff**, приховати в декількох контейнерах різного формату текстовий файл.
2. Використовуючи програму OpenPuff, витягти текстовий файл з набору контейнерів.
3. У звіті опишіть порядок виконання завдання.

А5. Впровадження графічного файлу у відеоконтейнер.

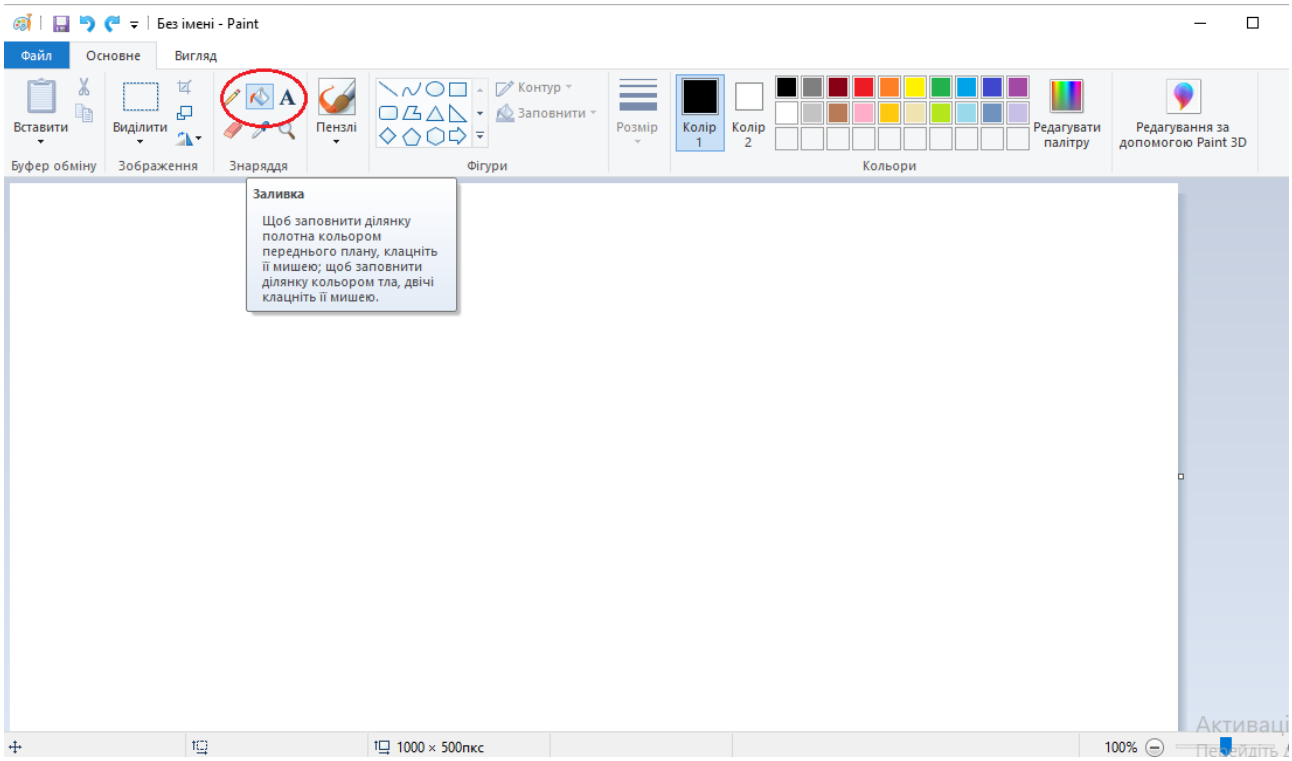
4. Використовуючи програму OpenPuff, приховати у відео контейнері формату mp4 графічний файл.
5. Використовуючи програму OpenPuff, витягти графічний файл з відеоконтейнера.
6. У звіті опишіть порядок виконання завдання.

Завдання Б. Напишіть програму, яка використовує стеганографічний алгоритм (LSB або інший) для приховування текстового повідомлення у графічному файлі.

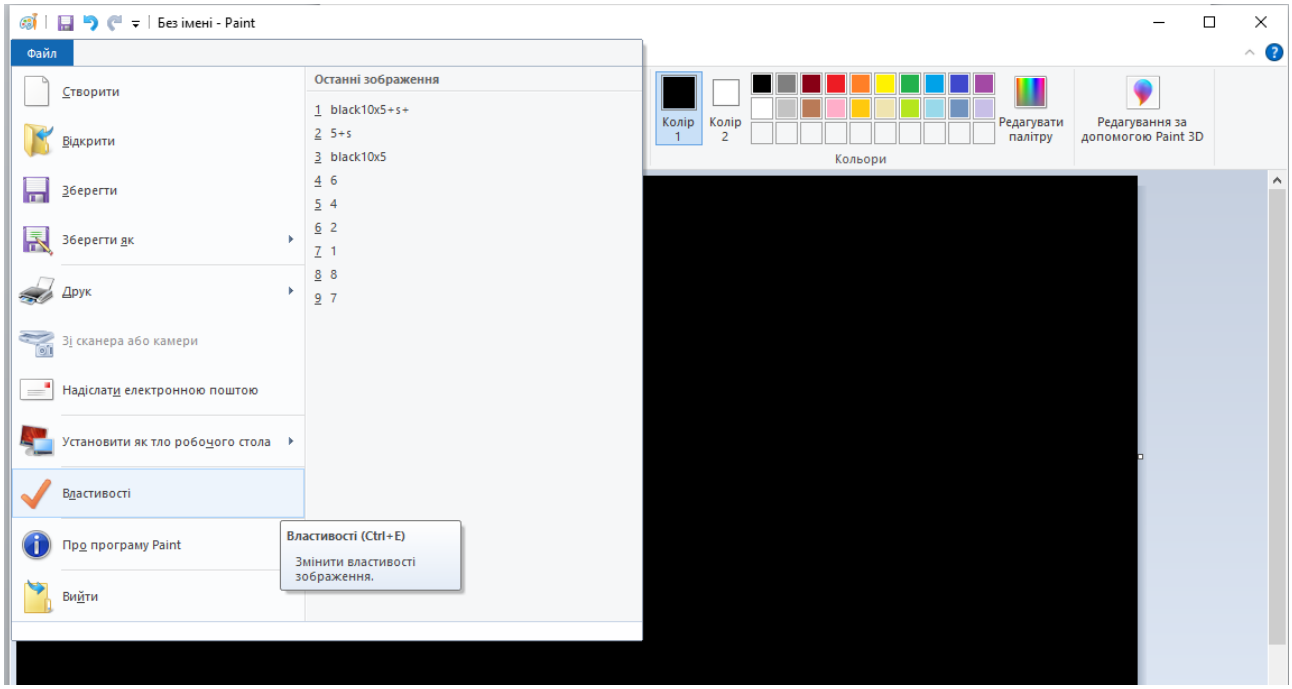
Додаток 1. Методичні вказівки до завдання А3.

7. У графічному редакторі MS Paint створити малюнок (прямокутник або квадрат) із заданими атрибутами і вказаним кольором заливки паперу. Для прямокутника спочатку вказана його ширина, а потім - висота. Наприклад, розмір 10x4 означає, що прямокутник має ширину: 10 точок (пікселів) і висоту 4 точки.

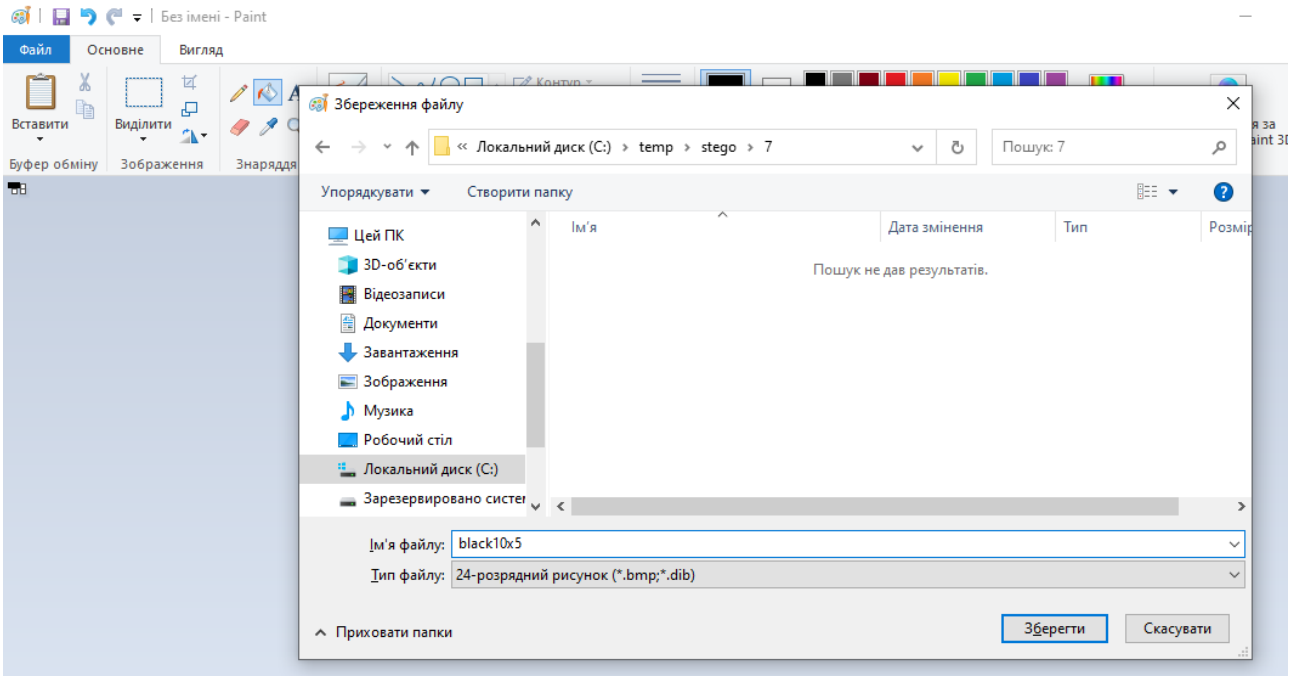
- Створимо чорний прямокутник розміром 10x5 пікселів. Для цього відкриємо програму MS Paint, вибравши інструмент **Заливка** замалюємо фон полотна чорним кольором.



- Виберемо меню **Файл->Властивості-> Пікселі**: Ширина – 10, Висота –5.

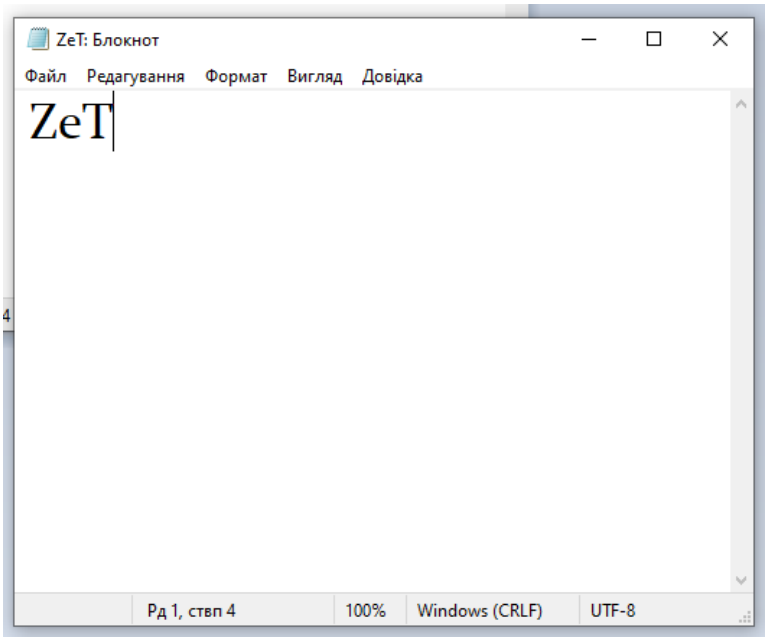


- Збережемо файл у форматі 24-розрядний рисунок BMP:

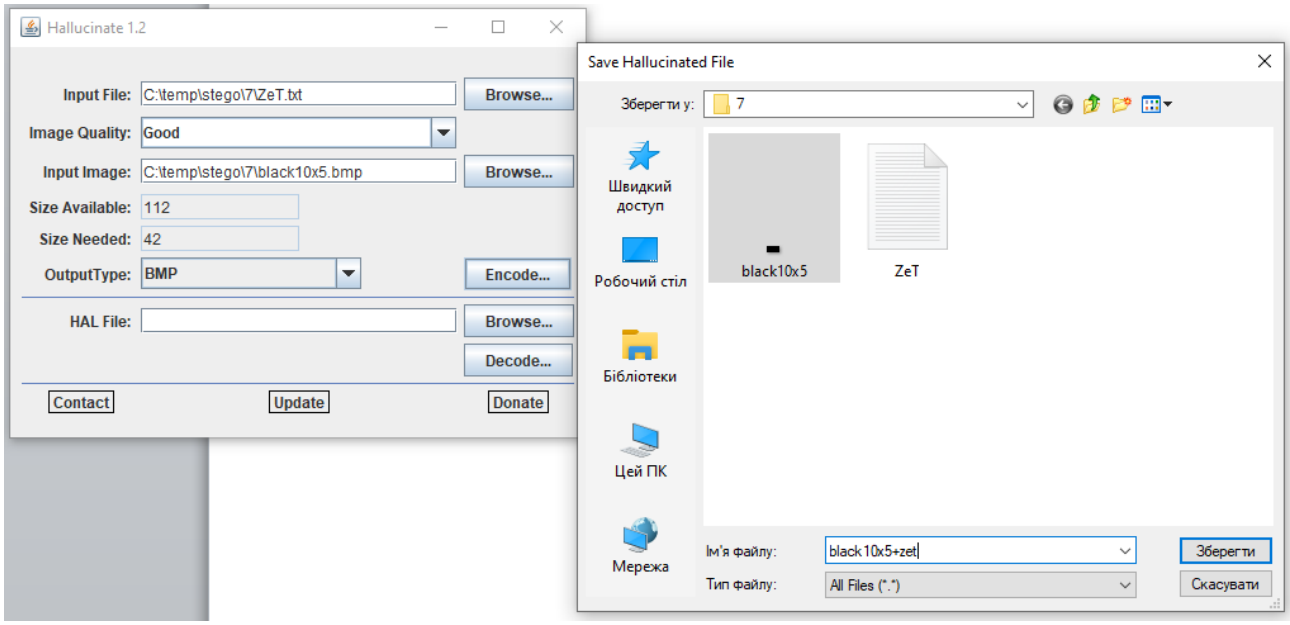


8. Впровадити в малюнок (помістити в контейнер) за допомогою програми Hallucinate заданий текст.

- Створимо текстовий файл в Блокноті із заданою фразою, наприклад, **ZeT.**



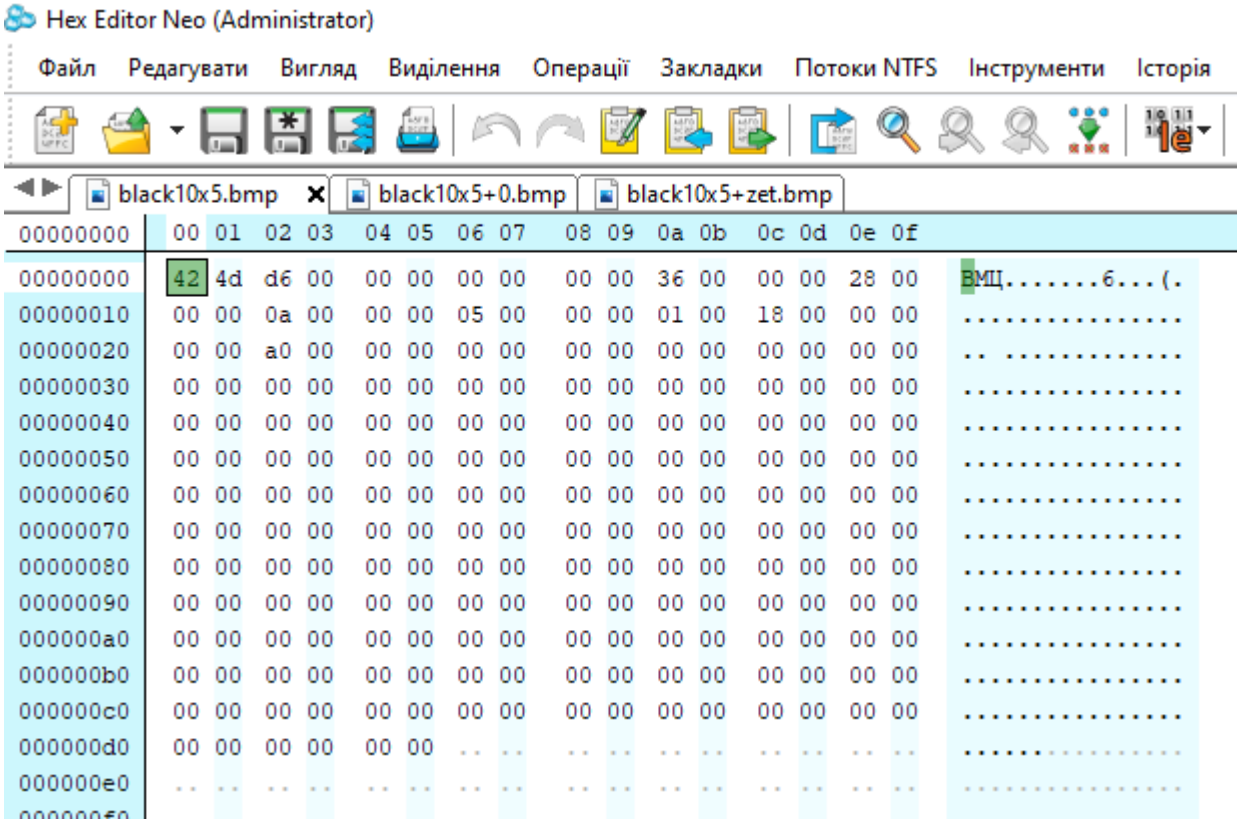
- Використовуючи програму Hallucinate впроваджуємо заданий текстовий файл у рисунок.



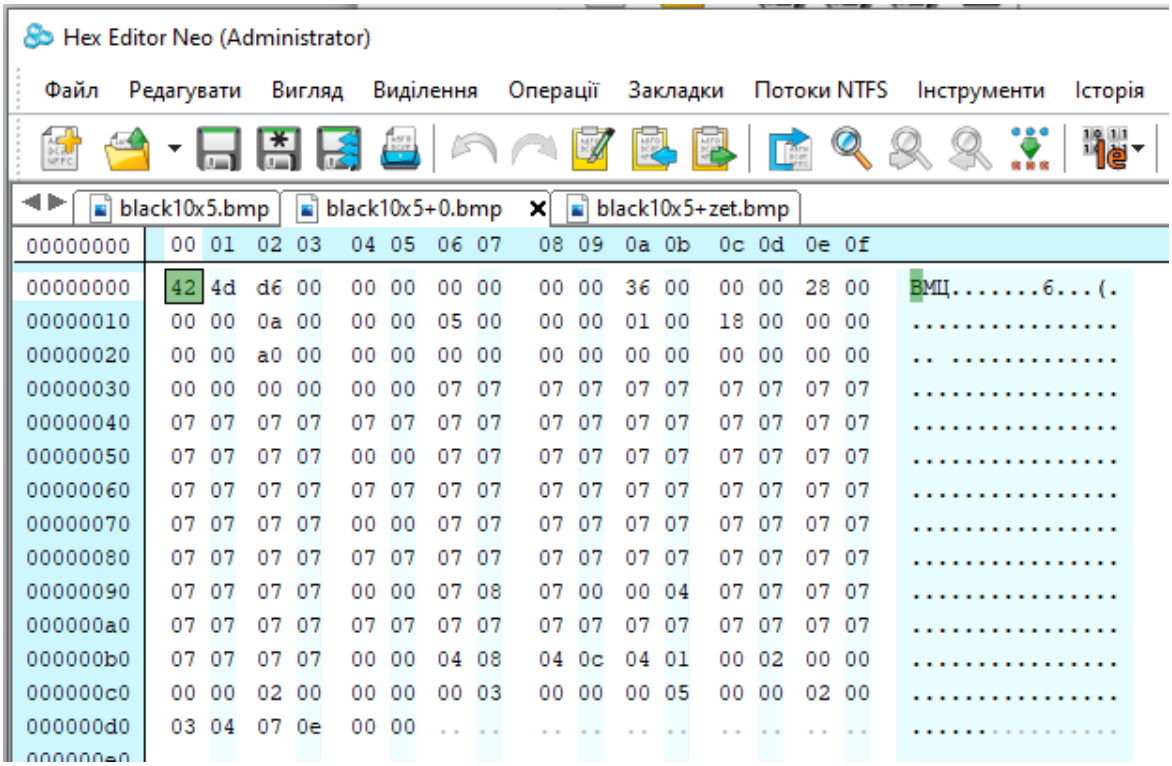
9. У звіті представити бітові карти малюнка без тексту і з текстом. Бітові карти слід отримати за допомогою програми Hex Editor Neo (https://www.hhdsoftware.com/free-hex-editor) або онлайн https://hexed.it/.

Оскільки ми впроваджуємо не лише три літери у графічний контейнер, а цілий текстовий файл (містить невидимі позначки кінець рядка, кінець файлу тощо), то бітова карта графічного файлу зміниться сильніше. Щоб побачити різницю можемо впровадити пустий текстовий файл у графічний контейнер.

Бітова карта пустого графічного файлу



Бітова карта контейнера BMP з пустим текстовим файлом



Бітова карта контейнера BMP з текстовим файлом ZeT