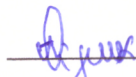


Міністерство освіти і науки України
Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки

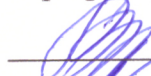
ПОЯСНЮВАЛЬНА ЗАПИСКА

до випускної кваліфікаційної роботи бакалавра
на тему: «Проект навчального набору завдань за напрямком Cryptography та Steganography для студентського змагального CTF-майданчика»

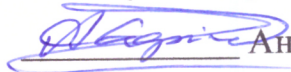
Виконала студентка 4-го курсу групи КБ-20-1
спеціальності 125 «Кібербезпека»

 Анастасія СІРИК

Керівник: професор кафедри КІ та КБ, д.т.н., доцент

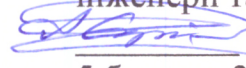
 Володимир ВОРОТНИКОВ

Рецензент: завідувач кафедри КІ та КБ, к.т.н., доцент

 Андрій ЄФІМЕНКО

Житомир – 2024

Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки
Спеціальність 125 «Кібербезпека»
Освітня програма «Кібербезпека»

ЗАТВЕРДЖУЮ
Завідувач
кафедри комп'ютерної
інженерії та кібербезпеки
 Андрій Єфіменко
5 березня 2024 р.

ЗАВДАННЯ

на випускн кваліфікаційну роботу бакалавра

Студентка: Сірик Анастасія Василівна

Тема роботи: «Проект навчального набору завдань за напрямком Cryptography та Steganography для студентського змагального CTF-майданчика»

затверджена Наказом університету від 5 березня 2024 р. №98

Термін здачі студентом закінченої роботи 9 червня 2024 р.

Вихідні дані роботи:

Об'єктом дослідження є процес розробки криптографічних та стеганографічних завдань для студентського змагального CTF-майданчика.

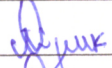
Предметом дослідження є навчальний набір різноманітних завдань різного рівня складності з криптографії та стеганографії для студентського змагального CTF-майданчика.

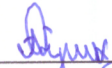
Обладнання та програмне забезпечення:

1. Операційна система Windows 11 Pro
2. Операційна система Kali Linux (2023.3)
3. Oracle VM VirtualBox (versions 7.0)
4. PyCharm (2023.3.3)

5. Файловий архіватор WinRAR
6. Редактор метаданих ExifTool
7. Графічний редактор GIMP (2.10.36)
8. Графічний редактор Microsoft Paint
9. Відеоредактор Microsoft Clipchamp
10. Графічним синтезатором звуку Coagula
11. Редактор звукових файлів Audacity
12. Аудіоаналізатором Sonic Visualiser
13. Стеганографічні програми та утиліти: Outguess, Openstego, Stegsnow, Steghide, DeepSound, QSSTV та інші.

Консультанти з випускної кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Покотило О.А.	 05.03.2024	 05.03.2024
2	Покотило О.А.	 05.03.2024	 05.03.2024
3	Покотило О.А.	 05.03.2024	 05.03.2024

Студентка  Анастасія Сірик

Керівник  Володимир Воротніков

Календарний план

№ з/п	Назви етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка завдання	5 березня 2024	Виконано
	Підготовка плану роботи	7 березня 2024	Виконано
2	Пошук інформації стосовно організації та проведення CTF-змагань	10 березня 2024 – 13 березня 2024	Виконано
3	Огляд криптографічних методів та алгоритмів	15 березня 2024 – 20 березня 2024	Виконано
4	Ознайомлення із різними видами стеганографії	21 березня 2024 – 27 березня 2024	Виконано
5	Складання списку літературних джерел	28 березня 2024	Виконано
6	Розробка та формулювання методики розробки завдань для CTF-змагань	1 квітня 2024 – 8 квітня 2024	Виконано
7	Аналіз різних інструментів для створення та розв'язань CTF – завдань	10 квітня 2024 – 22 квітня 2024	Виконано
8	Розробка та тестування криптографічних завдань	25 квітня 2024 – 9 травня 2024	Виконано
9	Реалізація й перевірка стеганографічних завдань	10 травня 2024 – 24 травня 2024	Виконано
10	Внесення коригувань та фіналізація завдань	25 травня 2024 – 31 травня 2024	Виконано
11	Написання пояснювальної записки	1 червня 2024 – 15 червня 2024	Виконано
12	Отримання відгуку та рецензії	18 червня 2024	Виконано
13	Оформлення презентації	19 червня 2024 – 22 червня 2024	Виконано
14	Захист	24 червня 2024	

Студентка  Анастасія Сірик

Керівник  Володимир Воротніков

АНОТАЦІЯ

Кваліфікаційна робота складається з проєкту навчального набору завдань за напрямком Cryptography та Steganography для студентського змагального CTF-майданчика. Проєкт спрямований на створення завдань, що охоплюють різні аспекти криптографії та стеганографії, з метою підвищення знань та навичок студентів у цих галузях.

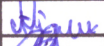
Пояснювальна записка містить 138 сторінок, включаючи 109 рисунків та 5 таблиць. Список використаних джерел налічує 43 найменувань та розміщений на 3 сторінках.

Метою випускної кваліфікаційної роботи є розробка навчального набору завдань для студентського CTF-майданчика, який дозволить учасникам на практиці застосовувати теоретичні знання з криптографії та стеганографії.

Основні результати роботи включають:

- Визначено ключові завдання для розробки навчального набору, відповідно до актуальних тем в криптографії та стеганографії;
- Проаналізовано базові методи і алгоритми, що використовуються в CTF-змаганнях;
- Розглянуто особливості створення завдань, що включають різні типи шифрування, методи приховування даних, та інструменти для їх аналізу;
- Показано розробку завдань різних рівні складності з докладним описом їхнього вирішення;
- Наведено результати тестування завдань.

Ключові слова: CTF, ЗМАГАННЯ, КРИПТОГРАФІЯ, СТЕГANOГРАФІЯ, ШИФРУВАННЯ, ІНФОРМАЦІЯ, ДАНІ, ЗОБРАЖЕННЯ, ТЕКСТ, ІНСТРУМЕНТИ, МЕТОДИ, АЛГОРИТМИ.

					ККІКБ.КР.Б – 125 – 18 – ПЗ		
Змн.	Арк.	№ докум.	Підпис	Дата			
Розроб.		Сірик А.В.		18.08.24	Літ.	Арк.	Аркушів
Керівник		Воротніков В.В.		18.08.24		5	154
Реценз.		Сфіменко А.А.		18.08.24	ФІКТ, Гр. КБ-20-1		
Н. контр.		Воротніков В.В.		18.08.24			
Затверд.		Сфіменко А.А.		18.08.24			

ANNOTATION

The qualification work consists of a project for developing a set of educational tasks in the field of Cryptography and Steganography for a student competition CTF platform. The project aims to create a comprehensive set of tasks covering various aspects of cryptography and steganography to enhance students' knowledge and skills in these areas.

The explanatory note contains 136 pages, including 109 figures and 5 tables. The list of references includes 43 titles and is placed on 3 pages.

The goal of the qualification work is to develop a set of educational tasks for a student CTF platform that will allow participants to apply theoretical knowledge in cryptography and steganography practically.

The main results of the work include:

- Identifying key topics for developing the educational set based on current trends in cryptography and steganography;
- Analysing basic methods and algorithms used in CTF competitions;
- Examining the features of creating tasks that include various types of encryption, data hiding methods, and tools for their analysis;
- Developing tasks of varying difficulty levels with detailed solution descriptions;
- Presenting the results of task testing.

Keywords: CTF, COMPETITION, CRYPTOGRAPHY, STEGANOGRAPHY, ENCRYPTION, INFORMATION, DATA, IMAGES, TEXT, TOOLS, METHODS, ALGORITHMS.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		6

ЗМІСТ

ВСТУП.....	11
РОЗДІЛ 1 АНАЛІЗ МЕТОДІВ Й ЗАСОБІВ КРИПТОГРАФІЇ ТА СТЕГАНОГРАФІЇ, ЩО ВИКОРИСТОВУЮТЬСЯ ДЛЯ STF-ЗМАГАНЬ	13
1.1 Організація та проведення змагань у форматі STF	13
1.2 Криптографічні методи та алгоритми для STF-змагань	17
1.2.1 Основні поняття криптографії.....	17
1.2.2 Класичні алгоритми шифрування	21
1.2.3 Симетричні алгоритми шифрування.....	27
1.2.3.1 Advanced Encryption Standard	29
1.2.4 Асиметричні алгоритми шифрування.....	30
1.2.4.1 Rivest–Shamir–Adleman.....	31
1.2.5 Візуальна криптографія	32
1.2.6 Кодування.....	33
1.2.7 Азбука Морзе	35
1.3 Стеганографічні методи та алгоритми для STF-змагань.....	36
1.3.1 Стеганографія в тексті	37
1.3.2 Стеганографія в зображеннях.....	41
1.3.3 Стеганографія в аудіо	47
1.3.4 Стеганографія у відео	49
Висновки до першого розділу	51
РОЗДІЛ 2 ПРОЕКТУВАННЯ ЗАВДАНЬ, АНАЛІЗ ТА ВИБІР ІНСТРУМЕНТІВ ДЛЯ ЇХ РЕАЛІЗАЦІЇ ТА РОЗВ’ЯЗАННЯ У STF-ЗМАГАННЯХ.....	53
2.1 Методика розробки завдань для STF-змагань.....	53
2.2 Аналіз інструментів для розробки та вирішення криптографічних завдань	60
2.2.1 Онлайн ресурси та інструменти	60
2.2.2 Бібліотеки Python для криптографії	62
2.2.3 RSA інструменти.....	64

2.3	Аналіз інструментів для розробки та вирішення стеганографічних завдань	65
2.3.1	Інструменти для зображень	65
2.3.2	Інструменти для аудіо та відео	67
2.3.3	Інструменти для текстових файлів.....	68
2.4	Базові інструменти для STF-завдань	70
2.4.1	Інструменти для початкового аналізу	70
2.4.2	Інструменти для роботи з архівами.....	71
	Висновки до другого розділу	73
РОЗДІЛ 3 ПРАКТИЧНА РЕАЛІЗАЦІЯ КРИПТОГРАФІЧНИХ ТА СТЕГANOГРАФІЧНИХ ЗАВДАНЬ ДЛЯ STF-ЗМАГАНЬ.....		74
3.1	Практична реалізація криптографічних завдань легкого рівня	74
3.1.1	«Julius Caesar»	74
3.1.2	«Lovely Hill Picture»	77
3.1.3	«Vigenère Quest».....	78
3.2	Практична реалізація криптографічних завдань середнього та складного рівнів	82
3.2.1	«D1v3 Into Th3 D3pths!»	82
3.2.2	«The Starry Bytes».....	88
3.2.3	«Rhythmic Symphony of Art»	90
3.2.4	«Modern Art».....	94
3.2.5	«Art Enthusiasts' Seminar»	96
3.3	Практична реалізація стеганографічних завдань легкого рівня	98
3.3.1	«The Winged Victory of Samothrace»	98
3.3.2	«Unveiling the Invisible Art»	100
3.3.3	«Black Square»	102
3.4	Практична реалізація стеганографічних завдань середнього та складного рівнів	105
3.4.1	«Mona Lisa»	105
3.4.2	«Silent Nights, Ode to Winter»	109

3.4.3	«Art in the Digital Age»	112
3.4.4	«Trojan Horse»	114
3.4.5	«Museum Advertising»	116
3.4.6	«Melodies of Life»	127
	Висновки до третього розділу	134
	ВИСНОВКИ	136
	СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	137
	ДОДАТКИ	140

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		9

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

CTF – Capture The Flag

RSA – Rivest, Shamir та Adleman

DES – Data Encryption Standard

AES – Advanced Encryption Standard

ASCII – American Standard Code for Information Interchange

UTF – Unicode Transformation Format

LSB – Least Significant Bit, найменший значущий біт

PNG – Portable Network Graphics

BMP – Bitmap

JPEG – Joint Photographic Experts Group

GIF – Graphics Interchange Format

EXIF – Exchangeable Image File Format

WAV – Waveform Audio Format

MP3 – MPEG Audio Layer III

MP4 – MPEG-4 Part 14

AVI – Audio Video Interleave

MOV – QuickTime File Format

FLV – Flash Video Format

PPM – Portable PixMap Format

SSTV – Slow-Scan Television

OCR – Optical Character Recognition, оптичне розпізнавання символів

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		10

ВСТУП

У сучасному світі інформаційні технології стрімко розвиваються, відкриваючи нові можливості та одночасно породжуючи загрози для безпеки даних. З кожним роком зростає кількість кібератак, витоків конфіденційної інформації та інших інцидентів, що підкреслює необхідність впровадження надійних механізмів захисту даних. Одним з найефективніших способів захисту інформації є застосування криптографії та стеганографії. Криптографія дозволяє шифрувати дані, зберігаючи їхню конфіденційність та цілісність, тоді як стеганографія надає можливість приховувати сам факт існування інформації. Обидва ці напрямки є надзвичайно важливими для забезпечення безпеки в цифровому середовищі.

Однак, для ефективного використання криптографії та стеганографії необхідні глибокі знання та практичні навички. Одним з методів навчання та підвищення кваліфікації спеціалістів у цій галузі є участь у змаганнях формату Capture The Flag (CTF).

Проект навчального набору завдань за напрямком Cryptography та Steganography для студентського змагального CTF-майданчика є актуальним та важливим кроком у підготовці майбутніх фахівців. Цей набір завдань не лише допоможе студентам здобути необхідні вміння та досвід, але й сприятиме розвитку креативного підходу до вирішення задач із захисту інформації.

Основною метою цієї роботи є розробка навчального набору завдань за напрямками криптографії та стеганографії для студентського змагального CTF-майданчика. Це дозволить студентам поглиблено вивчати теоретичні аспекти та вдосконалювати практичні навички у вирішенні викликів різної складності.

Для досягнення мети роботи необхідно вирішити такі завдання:

- Провести аналіз основних криптографічних алгоритмів та методів стеганографії;
- Ознайомитися із існуючими підходами та методиками розробки CTF-завдань за напрямками криптографії та стеганографії;
- Створити набір практичних завдань різного рівня складності для CTF-майданчика;

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		11

– Перевірити розроблені завдання та оцінити їх ефективність.

Об'єктом дослідження є процес розробки криптографічних та стеганографічних завдань для студентського змагального CTF-майданчика. Цей процес включає вивчення існуючих алгоритмів, методів та підходів, а також створення різноманітних завдань для практичного застосування.

Предметом дослідження є навчальний набір завдань з криптографії та стеганографії для студентського змагального CTF-майданчика. Цей набір включає різноманітні виклики, які мають різний рівень складності, що дозволяє студентам поступово підвищувати свої знання та досвід у цих областях.

У процесі виконання кваліфікаційної роботи було розроблено проект навчального набору завдань з криптографії та стеганографії для студентського змагального CTF-майданчика. Створені завдання мають високу практичну цінність і можуть бути використані у освітньому процесі для підготовки студентів до реальних викликів у сфері кібербезпеки. Задачі у наборі різноманітні за типами і складністю, що забезпечує всебічне вивчення теми та розвиток навичок. Використання цього комплекту сприятиме підвищенню рівня знань та умінь учасників, і це, в свою чергу, покращить їхню конкурентоспроможність на ринку праці. Впровадження цих завдань у освітні програми навчальних закладів дозволить якісно підготувати майбутніх фахівців у галузі інформаційної безпеки.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		12

РОЗДІЛ 1 АНАЛІЗ МЕТОДІВ Й ЗАСОБІВ КРИПТОГРАФІЇ ТА СТЕГАНОГРАФІЇ, ЩО ВИКОРИСТОВУЮТЬСЯ ДЛЯ СТФ-ЗМАГАНЬ

1.1 Організація та проведення змагань у форматі СТФ

У світі кібербезпеки постійно точиться запекла боротьба між зловмисниками та кіберзахисниками. Змагання Capture The Flag (CTF) пропонують захоплюючий спосіб не лише перевірити, а й удосконалити свої навички в цій динамічній сфері. Це не просто розваги, а й цікаві навчальні завдання, що імітують реальні сценарії. Учасникам доводиться шукати, використовувати та усувати вразливості, поринаючи у захоплюючий світ кібербезпеки.

Ці змагання, які часто проводяться онлайн, тривають від декількох годин до пари днів. Учасники можуть змагатися як індивідуально, так і в команді. Їхня мета – «захопити прапори», які в цьому випадку є віртуальними маркерами або фрагментами інформації, прихованими в системах чи мережах.

Існує декілька форматів СТФ, які роблять ці змагання цікавими та динамічними, а саме:

- Jeopardy - це формат, у якому учасники розв'язують окремі завдання з різних категорій та отримують за це бали;
- Attack-Defence (Напад-Захист) – це формат, у якому команди повинні одночасно захищати свої сервери від атак суперників і намагатися зламати системи інших команд [10];
- Mixed competitions (Змішаний формат) - комбінація обох форматів, де учасники виконують як окремі завдання (Jeopardy), так і беруть участь у командних змаганнях із захисту та атаки (Attack-Defence) [9];

Jeopardy – це формат, який пропонує командам серію завдань різної складності: від рівня easy/beginner (легкий) до hard/insane (складний). Завдання поділяються на різні категорії, такі як:

- веб-безпека;
- зворотне проектування;
- криптографія;

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		13

– стеганографія та інші.

Веб-безпека (Web Security) – це завдання, що зосереджені на виявленні та експлуатації вразливостей у веб-сайтах через використання протоколів і технологій, як-от PHP, CMS (наприклад, Django), SQL, JavaScript. Для їх розв'язання часто застосовують SQL-ін'єкції, XSS-атаки, CSRF-атаки та інші методи [10].

Зворотне проектування (Reverse Engineering) кидає виклик учасникам розкрити секрети скомпільованої (машинного коду, байт-коду) програми. Розв'язання таких завдань включає декомпіляцію коду, аналіз даних та інші методи для розуміння того, як працює програма.

У завданнях **криміналістики** (Forensics) учасникам потрібно досліджувати та відновлювати дані з цифрових артефактів, таких як жорсткі диски, USB-накопичувачі, файли захоплених мережевих пакетів. Розв'язання таких завдань включає пошук видалених файлів, аналіз метаданих, дамтів пам'яті та інші методи для збору доказів [10].

Бінарна експлуатація (Binary Exploitation) або Pwn – це завдання, де учасникам потрібно аналізувати та експлуатувати скомпільовані програми, наприклад, Linux ELF або Windows EXE для отримання контролю над ними. Це включає пошук можливих шляхів переповнень буфера, форматування коду та інші методи [10].

Завдання з **криптографії** (Cryptography) кидають виклик учасникам у розшифруванні повідомлень, використовуючи знання криптографічних алгоритмів та методів атак для злomu шифрів.

Стеганографія (Steganography) – це завдання, де учасникам потрібно відшукати приховану інформацію в медіафайлах, таких як зображення, аудіофайли, відеофрагменти та текстових файлах.

Різне (Miscellaneous, Misc) – це абсолютно випадкові та безпрецедентні завдання, для вирішення яких потрібна лише логіка, знання та терпіння. Вони можуть охоплювати всі вищезазначені категорії, поєднані їх між собою.

За кожне розв'язане завдання команда отримує певні бали. Зазвичай їхня кількість зростає зі збільшенням складності питання. Команди проходять завдання

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		14

послідовно, розкриваючи нові лише після успішного виконання попередніх (іноді можуть бути доступні відразу).

Переможцем стає команда, яка набрала найбільше балів за визначений проміжок часу. Відомими прикладами CTF-змагань у форматі Jeopardy є:

- Hack The Box CTF;
- PicoCTF;
- Sikorsky CTF.

Hack The Box – це популярна онлайн-платформа для навчання та розвитку навичок у сфері кібербезпеки, яка регулярно проводить CTF-змагання. Вони пропонують різноманітні завдання, що охоплюють такі теми, як експлуатація вразливостей, зворотне проектування, веб-безпека та криптографія. Змагання Hack The Box CTF відомі своєю високою якістю завдань та участю великої кількості команд з усього світу.

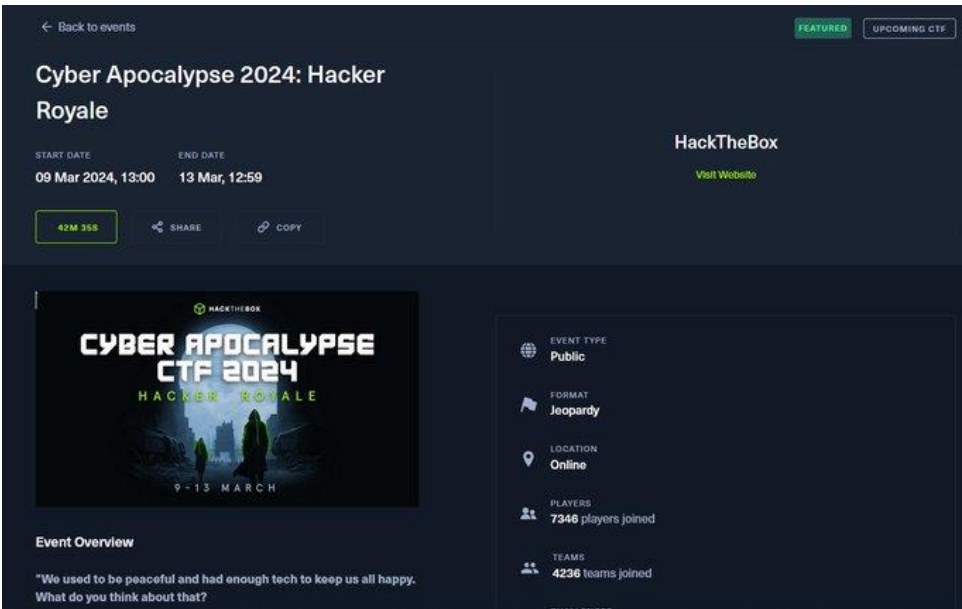


Рисунок 1.1 – Змагання Hack The Box CTF

PicoCTF – це змагання з кібербезпеки, організоване Університетом Карнегі-Меллон. Воно спеціально розроблене для учнів середніх та старших класів, проте участь можуть брати всі бажаючі. PicoCTF пропонує широкий спектр завдань різної складності, які допомагають учасникам здобути базові знання та навички у сфері інформаційної безпеки. Змагання включають задачі з криптографії, програмування, мережевої безпеки та інших областей.

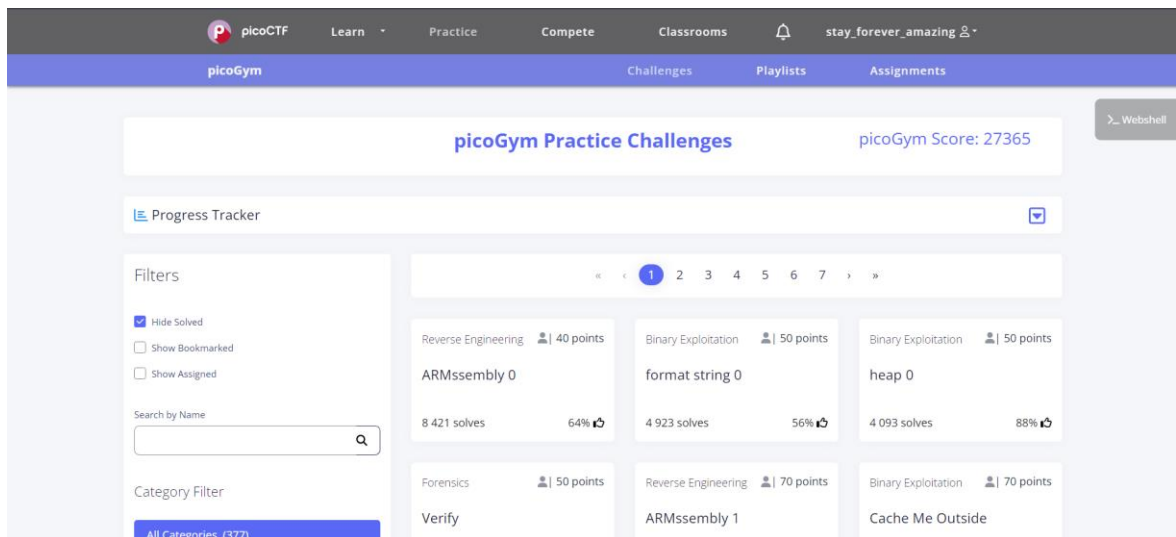


Рисунок 1.2 – Вигляд платформи PicoCTF

Sikorsky CTF – це щорічне змагання, організоване проектом USAID «Кібербезпека критично важливої інфраструктури України» у партнерстві з КПП ім. Ігоря Сікорського та Amazon Web Services (AWS). Змагання проводяться серед студентів вишів та старшокласників (учнів 9-11 класів) українських шкіл для розвитку кадрового потенціалу України у сфері кібербезпеки. Завдання охоплюють різноманітні аспекти кібербезпеки, включаючи криптографію, веб-безпеку, зворотне проектування та аналіз шкідливого програмного забезпечення.

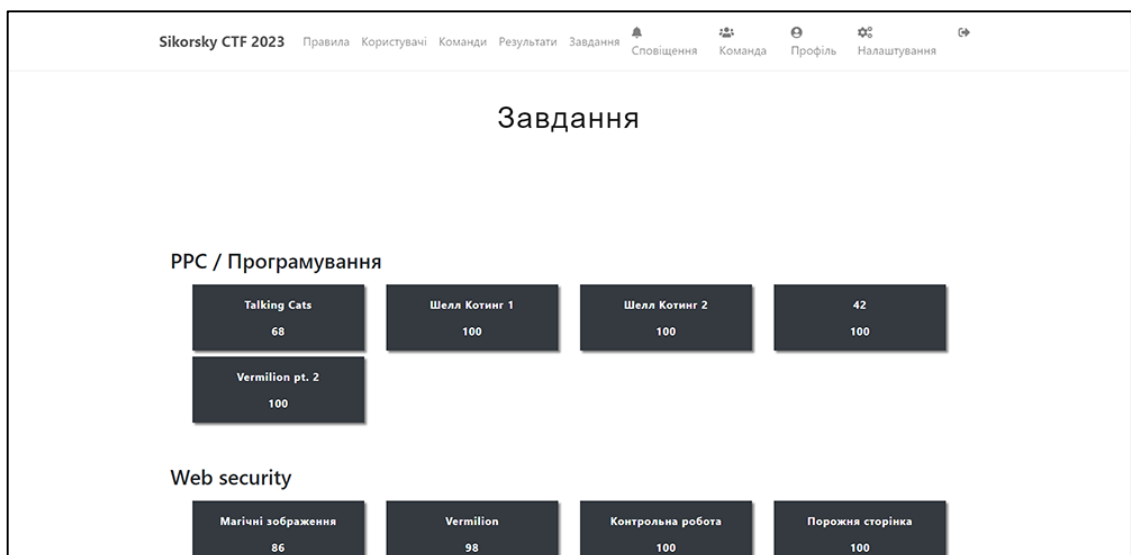


Рисунок 1.3 – Змагання Sikorsky CTF

Існують ще інші змагання у форматі Jeopardy, які також є популярними та престижними, наприклад TAMUctf, Google CTF, BSidesSF CTF, sqlrrel CTF, WolvCTF, UofTCTF та інші.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				16
Змн.	Арк.	№ докум.	Підпис	Дата		

Attack-Defence - це формат, що кидає виклик учасникам у їхніх здібностях як захищати, так і атакувати. Кожна команда отримує власну мережу або хост з уразливими сервісами. Їм надається час на те, щоб укріпити свої системи та розробити експлойти для атаки на мережі інших команд. Потім вони змагаються, намагаючись зламати одне одну, захищаючи при цьому власні сервери. Цей формат CTF потребує стратегії, командної роботи та гнучкості. DEF CON CTF - відомий приклад таких змагань.

Mixed competitions - це формат, що поєднує елементи Jeopardy та Attack-Defence, пропонуючи унікальні та динамічні змагання. Наприклад, UCSB iCTF використовує формат «військової гри» з певними часовими рамками для виконання завдань [9].

Як і на багатьох інших змаганнях, рівень навичок для CTF змінюється в залежності від події. Деякі з них орієнтовані на професіоналів із досвідом роботи в сфері кібербезпеки. Такі змагання зазвичай мають значні грошові винагороди та можуть проводитися на базі великих компаній. Інші ж призначені для студентів шкіл, коледжів та університетів, іноді надаючи грошову підтримку для навчання тим, хто виявляє високий рівень знань.

1.2 Криптографічні методи та алгоритми для CTF-змагань

1.2.1 Основні поняття криптографії

Захист цінної інформації є потребою, що існує віддавна. Ще в часи Римської імперії Юлій Цезар шифрував повідомлення за допомогою криптографічних засобів. З розвитком телекомунікаційних систем криптографія стала науковою дисципліною, особливо після публікації Клодом Шенноном у 1949 році статті «Теорія зв'язку в секретних системах» [4].

Значний вплив на розвиток криптографії мала також стаття «Нові напрямки в криптографії» Вітфілда Діффі та Мартина Хеллмана 1976 року. Вони показали, що можливо забезпечити безпечний обмін інформацією через незахищені канали без спільного секретного ключа. Це відкриття сприяло появі криптографії з відкритим ключем і започаткувало нову дисципліну – інформаційну безпеку.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		17

Криптографія та криптологія тісно пов'язані, але різняться за своїм фокусом:

Криптографія – це наука та мистецтво створення методів для захисту інформації. Вона використовує математичні алгоритми для шифрування даних, перетворюючи відкритий текст у шифротекст, який можуть розшифрувати лише уповноважені сторони.

Криптологія – вивчення шифрів і криптосистем, що охоплює як криптографію (створення шифрів), так і криптоаналіз (розшифровка шифрів без знання ключа). Криптоаналітики знаходять слабкі місця в криптосистемах для отримання доступу до захищеної інформації.

В історії криптології умовно можна виділити чотири етапи:

- Наївний етап (до початку XVI ст.);
- Формальний етап (кінець XV — початок XX ст.);
- Науковий етап (1930-1960-ті рр.);
- Комп'ютерний етап (з 1970-х рр.).

Наївний етап (до початку XVI ст.)

Наївна криптологія характеризується використанням простих і примітивних методів заплутування змісту текстів, щоб ускладнити їх розуміння. На цьому етапі використовувалися кодування та стеганографія, які споріднені з криптологією, але не є тотожними їй. Більшість шифрів базувалися на перестановці або моноалфавітній підстановці [11]. Прикладами є шифр Цезаря та Полібійський квадрат. Ці методи були ефективними для свого часу, але стали вразливими з розвитком математики та комп'ютерних наук.

Формальний етап (кінець XV — початок XX ст.)

На цьому етапі з'явилися формалізовані шифри, стійкі до ручного криптографічного аналізу. В епоху Відродження в Європі розвиток науки та торгівлі викликав потребу в надійних методах захисту інформації [11].

Важливі фігури та події:

- Леон Баттіста Альберті запропонував багатоалфавітну підстановку та зробив шифр Віженера, який полягає в послідовному «складанні» літер початкового тексту з ключем;

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		18

- Йоганн Трисемус написав «Поліграфія» (1508 р.), де узагальнив і систематизував відомі на той час алгоритми шифрування;
- Шифр Плейфера розроблений на початку XIX ст. Чарльзом Уїтстоном;
- Огюст Керкгоффс сформулював принцип, що секретність шифрів повинна базуватися на секретності ключа, а не алгоритму;
- Роторні криптографічні системи отримали широке застосування на початку XX ст. і стали вершиною формальної криптографії. Однією з перших використовуваних машин стала німецька Enigma, розроблена 1917 р. Едвардом Хеберном і вдосконалена Артуром Кірхом.

Науковий етап (1930-1960-ті рр.)

Поява криптографічних систем зі строгим математичним обґрунтуванням їх стійкості [11].

Основні досягнення:

- Клод Шеннон написав «Теорія зв'язку в секретних системах» (1949 р.), яка заклала наукову основу криптографії та криптографічного аналізу;
- Сформувалися розділи математики, які стали основою криптології, зокрема теорія ймовірності, математична статистика, загальна алгебра, теорія чисел, теорія алгоритмів, теорія інформації, кібернетика.

Комп'ютерний етап (з 1970-х рр.)

Цей етап став можливим завдяки обчислювальним засобам із достатньою продуктивністю та потужністю для реалізації складних криптографічних систем [11].

Ключові події:

- Блокові шифри: перші практичні реалізації, такі як американський стандарт шифрування Data Encryption Standard (1978 р.);
- Асиметричні криптосистеми: з'явилися в середині 1970-х рр., що дозволило обмінюватися зашифрованою інформацією без передачі секретного ключа. Важливою віхою стала праця Уїтфілда Діффі та Мартіна Хеллмана (1976 р.) й відкриття системи RSA (Рон Рівест, Аді Шамір і Леонард Адлеман).

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		19

- Нові напрями: у 1980-90-х рр. з'явилися ймовірнісне шифрування, квантова криптографія та інші. В цей період було розроблено нефейстелейвські шифри (SAFER, RC6 та ін.), а у 2000 р. було прийнято новий стандарт шифрування AES, який замінив DES.

Одним з ключових принципів сучасної криптографії є принцип Керкгоффа, сформульований Огюстом Керкгоффсом наприкінці XIX століття. Правило стверджує, що безпека та стійкість криптографічної системи має залежати від секретності ключа, тоді як алгоритм може бути загальновідомим.

Криптосистема - це математична модель або алгоритм для забезпечення конфіденційності та безпеки інформації під час передачі чи зберігання. Вона включає методи шифрування (перетворення відкритого тексту в зашифрований текст) та дешифрування (зворотне перетворення) [3].

Базові компоненти криптосистеми:

- **Відкритий текст** (plaintext або cleartext) - зрозуміла інформація, яку можна прочитати без спеціальних знань чи інструментів;
- **Зашифрований текст** (ciphertext) – перетворений відкритий текст, який неможливо прочитати без спеціальних знань або інструментів. Іншими словами, це незрозуміла інформація, яку можна розшифрувати лише за допомогою відповідного ключа або алгоритму;
- Алгоритми **шифрування** (encryption) та **дешифрування** (decryption) – методи для перетворення інформації;
- **Ключ** (key) - це секретна інформація, яка використовується криптографічним алгоритмом для шифрування, розшифрування.

Існують два основні види криптографічних систем:

- Симетричні криптографічні системи;
- Асиметричні криптографічні системи.

Симетричні криптосистеми використовують один секретний ключ для шифрування та дешифрування. Це робить процес простим, але потребує безпечного обміну ключем між сторонами.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		20

Асиметричні криптосистеми використовують пару ключів: відкритий ключ для шифрування та секретний ключ для дешифрування. Це дозволяє безпечно обмінюватися інформацією без необхідності обміну секретним ключем.

1.2.2 Класичні алгоритми шифрування

Класичні алгоритми шифрування - це методи шифрування, які були розроблені та використовувалися до появи сучасних криптографічних систем. Ці алгоритми є досить простими у реалізації та розумінні тому, що вони базуються на стандартних математичних операціях та перетвореннях. Основні класичні алгоритми шифрування включають шифри заміни, перестановки та комбіновані шифри.

Шифр Цезаря

Шифр Цезаря - це один з найпростіших та найвідоміших класичних шифрів, який використовується для заміни літер у тексті зсувом на певну кількість позицій у алфавіті [4]. Цей шифр отримав свою назву на честь римського імператора Юлія Цезаря, який використовував подібний метод для забезпечення конфіденційності своїх повідомлень.

Основні принципи роботи шифру Цезаря:

- 1) Зсув літер. Кожна літера в алфавіті зсувається на фіксовану кількість позицій. Наприклад, якщо взяти зсув на 3 позиції, то літера А буде замінена на D, В на Е, С на F і так далі.

Таблиця 1.1 – Приклад зсуву літер

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
D	E	F	G	H	I	J		K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

- 2) Модульність алфавіту. Якщо зсув виходить за межі алфавіту (наприклад, зсув на 3 позиції для літери X), то зміщення знову починається з початку алфавіту. Тобто X замінюється на A, Y на B, Z на C.
- 3) Шифрування і дешифрування. Для шифрування тексту за допомогою шифру Цезаря кожна літера тексту замінюється на літеру, яка відповідає

зсуву в алфавіті. Для дешифрування зміщення здійснюється у зворотньому напрямку.

Формула шифрування для шифру Цезаря виглядає так:

$$E(x) = (x + k) \bmod n$$

де $E(x)$ – зашифрована літера;

x – позиція літери в алфавіті (наприклад, A = 0, B = 1, ..., Z = 25 для англійського алфавіту);

k – зсув, тобто кількість позицій, на які треба перемістити літеру вправо;

n – розмір алфавіту (наприклад, для англійського алфавіту $n = 26$).

Тоді формула дешифрування:

$$D(y) = (y - k) \bmod n$$

де $D(y)$ – зашифрована літера;

y – позиція зашифрованої літери в алфавіті (аналогічно, A = 0, B = 1, ..., Z = 25 для англійського алфавіту);

k – зсув, тобто кількість позицій, на які було переміщено зашифровану літеру вправо під час шифрування;

n – розмір алфавіту.

Наприклад, зашифруємо слово «CRYPTOGRAPHY» з використанням шифру Цезаря з зсувом на 3 позиції.

Таблиця 1.2 – Виконання шифрування

Відкритий текст	Зсув на 3 позиції	Зашифрований текст
C	$(2+3) \bmod 26 = 5$	F
R	$(17+3) \bmod 26 = 20$	U
Y	$(24+3) \bmod 26 = 1$	B
P	$(15+3) \bmod 26 = 18$	S
T	$(19+3) \bmod 26 = 22$	W
O	$(14+3) \bmod 26 = 17$	R
G	$(6+3) \bmod 26 = 9$	J
R	$(17+3) \bmod 26 = 20$	U
A	$(0+3) \bmod 26 = 3$	D
P	$(15+3) \bmod 26 = 18$	S
H	$(7+3) \bmod 26 = 10$	K
Y	$(24+3) \bmod 26 = 1$	B

Таким чином, зашифроване слово «CRYPTOGRAPHY» шифром Цезаря з зсувом на 3 позиції буде «FUBWSRUDSKB».

Шифр Цезаря має кілька переваг: по-перше, він простий у реалізації завдяки використанню елементарних арифметичних операцій, таких як додавання або віднімання, що дозволяє швидко шифрувати та розшифровувати дані. По-друге, ключ шифру Цезаря - це просто число, що визначає зсув літер у тексті, що робить його легким для запам'ятовування та обміну між сторонами. Нарешті, шифр часто використовують у навчальних цілях для ознайомлення студентів із основами шифрування та криптографії.

Основним недоліком шифру Цезаря – це вразливість до криптоаналізу, оскільки зсув є статичним і його легко виявити. Тому в сучасній криптографії шифр Цезаря використовується як простий приклад і не вважається надійним для захисту конфіденційності даних.

Шифр Віженера

Протягом приблизно 1500 років шифр Цезаря зазнав значних вдосконалень, трансформуючись у шифр Віженера, який був створений у 16 столітті італійським криптографом Джованом Баттіста Белласо. Назва шифру походить від французького криптографа Блеза де Віженера, який винайшов свій шифр у той же період. Однак через деякі неточності в історії шифр став відомим під іменем Віженера [5].

Незважаючи на це, шифр став популярним і знайшов застосування в різних історичних подіях, таких як громадянська війна в США, де його використовувала Конфедерація, а також у Швейцарії під час Першої світової війни.

Шифр Віженера є покроковим поліалфавітним шифром, який використовує ключове слово для послідовного зсуву літер у тексті.

Основні характеристики шифру Віженера:

- Ключове слово використовується для шифрування і дешифрування повідомлення. Воно повторюється до довжини тексту. Наприклад, якщо ключове слово - "KEY", а текст - "CRYPTOGRAPHY", то ключове слово буде повторене до "KEYKEYKEYKEY", і кожна літера тексту буде шифруватися за допомогою відповідної літери ключа;

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		23

- Шифр Віженера використовує кілька алфавітів для шифрування. Кожна літера відкритого тексту шифрується з використанням різних зсувів, залежно від літери ключа;
- Для шифрування символу x_i за допомогою ключа k_i використовується формула:

$$E(x_i) = (x_i + k_i) \bmod n$$

де $E(x_i)$ – зашифрований символ;

y_i – символ відкритого тексту;

k_i – символ ключа;

n – розмір алфавіту.

Повідомлення - CRYPTOGRAPHY зашифруємо ключем KEY.

Таблиця 1.3 – Виконання шифрування

C	R	Y	P	T	O	G	R	A	P	H	Y
K	E	Y	K	E	Y	K	E	Y	K	E	Y
2	17	24	15	19	14	6	17	0	15	7	24
10	4	24	10	4	24	10	4	24	10	4	24
12	21	22	25	23	12	16	21	24	25	11	22
M	V	W	Z	X	M	Q	V	Y	Z	L	W

В результаті отримуємо шифротекст MVWZXMQVYZLW.

Для дешифрування зашифрованого символу y_i за допомогою ключа k_i використовується формула:

$$D(y_i) = (y_i - k_i) \bmod n$$

де $D(y_i)$ – зашифрований символ;

y_i – символ відкритого тексту;

k_i – символ ключа;

n – розмір алфавіту.

Для зашифрування може використовуватися й таблиця Віженера або квадрат Віженера, є матрицею розміром 26×26 , де кожен рядок і стовпець мають заголовки від А до Z. Перший рядок цієї таблиці включає всі 26 англійських літер. Починаючи з другого рядка, кожен наступний рядок циклічно зсуває літери на одну позицію

вліво. Наприклад, коли буква В переміщується на перше місце у другому рядку, літера А знаходиться в кінці цього рядка.

Алфавіт повідомлення																													
	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			
А ф а в і т к л ю ч а	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			
	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	A			
	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B			
	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C			
	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D			
	F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E		
	G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F		
	H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G		
	I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H		
	J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I		
	K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J		
	L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K		
	M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L		
	N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M		
	O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N		
	P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O		
	Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P		
	R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q		
	S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R		
	T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S		
	U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T		
	V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U		
	W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V		
	X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W		
	Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X		
	Z	Z	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		

Рисунок 1.4 – Таблиця Віженера

Під час дешифрування потрібно спочатку знайти у першому стовпці таблиці Віженера літеру ключа, а потім переглянути літери шифротексту в тому ж рядку, щоб визначити, в якому стовпчику зверху знаходиться буква відкритого тексту.

Узагальнюючи, шифр Віженера вважається безпечнішим в порівнянні з шифром Цезаря через використання ключа з періодичністю та поліалфавітності. Однак він все ще вразливий до певних атак, таких як атака методом Казіскі. У сучасних системах криптографії шифр Віженера часто використовується як один із компонентів складніших шифрів або як демонстраційний приклад.

Шифр Гілла

Шифр Гілла, винайдений Лестером Гіллом у 1929 році, є поліграмним шифром підстановки, що ґрунтується на принципах лінійної алгебри. На відміну від шифрів моноалфавітного заміщення, де кожна буква алфавіту замінюється фіксованою іншою буквою, шифр Гілла оперує блоками тексту, що робить його стійкішим до злому.

Алгоритм використовує квадратну матрицю розміру $n \times n$, де n – це довжина блоку тексту, який буде шифруватися. Елементи матриці мають бути цілими числами, що обертаються за модулем 26 (тобто залишок від ділення на 26 має бути в діапазоні від 0 до 25). Ця матриця і є ключем шифру, який тримається в секреті [3].

Нехай ключова матриця буде такою: $K = \begin{pmatrix} 6 & 19 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{pmatrix}$

Початковий текст розбивається на блоки фіксованої довжини n . Блок тексту перетворюється на вектор n , де кожне число відповідає певному символу в алфавіті (наприклад, A=0, B=1, ..., Z=25).

$$P_1 = \begin{pmatrix} C \\ R \\ Y \end{pmatrix} = \begin{pmatrix} 2 \\ 17 \\ 24 \end{pmatrix} \quad P_2 = \begin{pmatrix} P \\ T \\ O \end{pmatrix} = \begin{pmatrix} 15 \\ 19 \\ 14 \end{pmatrix} \quad P_3 = \begin{pmatrix} G \\ R \\ A \end{pmatrix} = \begin{pmatrix} 6 \\ 17 \\ 0 \end{pmatrix} \quad P_4 = \begin{pmatrix} P \\ H \\ Y \end{pmatrix} = \begin{pmatrix} 15 \\ 7 \\ 24 \end{pmatrix}$$

Далі кожен вектор блоку множиться на матрицю ключа за модулем алфавіту. Отриманий вектор чисел знову перетворюється на блок тексту, де число відповідає зашифрованому символу.

$$K \cdot P_1 = \begin{pmatrix} 6 & 19 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{pmatrix} \cdot \begin{pmatrix} 2 \\ 17 \\ 24 \end{pmatrix} = \begin{pmatrix} 359 \\ 538 \\ 689 \end{pmatrix} \bmod 26 = \begin{pmatrix} 21 \\ 18 \\ 13 \end{pmatrix} = VSN$$

$$K \cdot P_2 = \begin{pmatrix} 6 & 19 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{pmatrix} \cdot \begin{pmatrix} 15 \\ 19 \\ 14 \end{pmatrix} = \begin{pmatrix} 465 \\ 639 \\ 833 \end{pmatrix} \bmod 26 = \begin{pmatrix} 23 \\ 15 \\ 1 \end{pmatrix} = XPB$$

$$K \cdot P_3 = \begin{pmatrix} 6 & 19 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{pmatrix} \cdot \begin{pmatrix} 6 \\ 17 \\ 0 \end{pmatrix} = \begin{pmatrix} 359 \\ 350 \\ 409 \end{pmatrix} \bmod 26 = \begin{pmatrix} 21 \\ 12 \\ 19 \end{pmatrix} = VMT$$

$$K \cdot P_4 = \begin{pmatrix} 6 & 19 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{pmatrix} \cdot \begin{pmatrix} 15 \\ 7 \\ 24 \end{pmatrix} = \begin{pmatrix} 247 \\ 547 \\ 779 \end{pmatrix} \bmod 26 = \begin{pmatrix} 13 \\ 1 \\ 25 \end{pmatrix} = NBZ$$

Отже, зашифроване слово «CRYPTOGRAPHY» за допомогою ключової матриці K буде «VSNXPBVMTNBZ».

Для дешифрування необхідно мати матрицю, обернену до ключової, і виконати аналогічні математичні операції.

Шифр Гілла має деякі переваги, такі як стійкість до багатьох методів криптоаналізу, особливо при великих розмірах ключової матриці. Однак він також має

свої обмеження, зокрема, розмірність ключа повинна бути квадратною та детермінованою.

1.2.3 Симетричні алгоритми шифрування

Симетрична криптосистема працює за допомогою наступних двох перетворень:

шифрування: $c = e_k(m)$,

дешифрування: $m = d_k(c)$,

де m – відкритий текст;

e – алгоритм шифрування;

d – алгоритм дешифрування;

k – секретний ключ;

c – зашифрований текст.

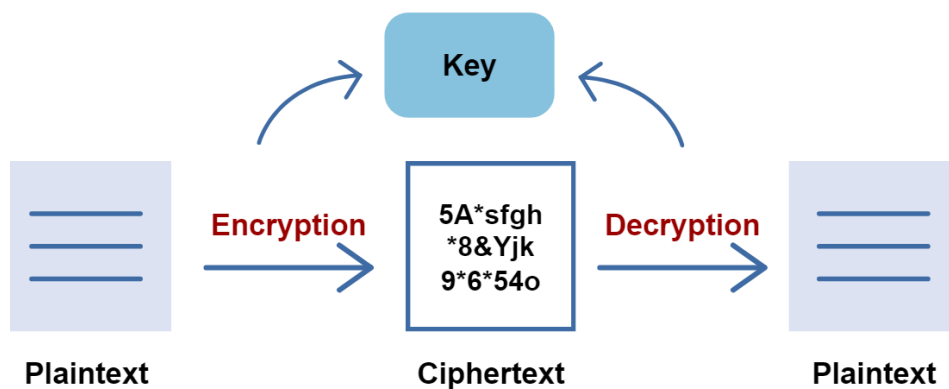


Рисунок 1.5 – Схема симетричного шифрування

У даному випадку, стійкість зашифрованої інформації залежить від секретності ключа k , адже методи шифрування та дешифрування можуть бути відомі всім.

Існує два основних типи шифрів для симетричного шифрування: блокові та потокові [1].

Блоковий шифр

Блоковий шифр шифрує блоки даних за допомогою ключа. Шифрування E генерує зашифрований блок C з відкритого тексту P та ключа K : $C=E(K,P)$. Дешифрування D відновлює відкритий текст P із зашифрованого блоку C : $P=D(K,C)$ [5].

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		27

Основні характеристики блокового шифру — розмір блоку та ключа. Наприклад, DES має блоки розміром 64 біти, AES — 128 бітів. Вибір конкретної довжини блоку залежить від потреб зашифрування, зберігання даних та вимог до безпеки системи. Важливо уникати занадто великих або малих значень для запобігання атакам.

Щоб забезпечити безпеку блокового шифру, використовуються різні техніки побудови раундів шифрування, такі як мережі заміни-перестановки (AES) та схеми Фейстеля (DES). Раундові ключі генеруються з основного ключа за допомогою спеціальних алгоритмів.

Блокові шифри використовують різні режими роботи для обробки довільних даних, зокрема: ECB (Electronic Codebook), CBC (Cipher Block Chaining), CFB (Cipher Feedback), OFB (Output Feedback), і CTR (Counter). Кожен режим має свої особливості щодо шифрування блоків і керування вектором ініціалізації. Найвідоміші блокові шифри - AES (Advanced Encryption Standard), DES (Data Encryption Standard), Triple DES (3DES), і Blowfish.

Потоковий шифр

Потоковий шифр шифрує дані біт за бітом або байтом з використанням ключа та генератора псевдовипадкової послідовності. Генератор псевдовипадкової послідовності (PRNG) створює ключі, які використовуються для шифрування та дешифрування. Формули шифрування та дешифрування базуються на операціях побітового XOR (ексклюзивного «або») між бітами відкритого тексту і бітами ключа [5].

Шифрування:

- Відкритий текст представлений як послідовність бітів $P = p_1 p_2 \dots p_n$;
- Ключ також послідовність бітів $K = k_1 k_2 \dots k_n$, де n - кількість бітів у відкритому тексті;
- Шифротекст $C = P \oplus K$, де $\oplus$ – операція XOR.

Дешифрування:

- Відкритий текст $P = C \oplus K$.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		28

Таблиця 1.4 – Операція XOR над бітами

$\oplus$	0	1
0	0	1
1	1	0

Однією з ключових проблем поточкових шифрів є необхідність синхронізації між відправником і отримувачем. Якщо один біт зміниться або буде втрачено під час передачі, це може призвести до втрати синхронізації і неправильного розшифрування даних. Поточкові шифри повинні бути захищеними від різних атак за допомогою надійних генераторів псевдовипадкових послідовностей та складних алгоритмів шифрування.

Популярні поточкові шифри включають RC4, ChaCha, Salsa20 та інші, кожен з яких має свої особливості та рекомендований для різних застосувань з урахуванням потреб у швидкості, безпеці й ресурсах обчислювальної системи.

1.2.3.1 Advanced Encryption Standard

Advanced Encryption Standard (AES), відомий також як Rijndael — це симетричний алгоритм шифрування, який широко використовується у всьому світі для захисту даних. Він був встановлений Національним інститутом стандартів і технологій США (NIST) у 2001 році як заміна DES через вразливості та обмеження останнього. AES став фактичним стандартом шифрування в різних додатках, включаючи державний, військовий та комерційний сектори [2].

AES обробляє блоки розміром 128 біт, використовуючи секретний ключ довжиною 128, 192 або 256 біт. Найпоширенішим є 128-бітний ключ, оскільки він робить шифрування дещо швидшим і різниця між 128- і 256-бітним захистом не має сенсу для більшості програм.

На відміну від деяких шифрів, які працюють з окремими бітами або 64-бітними словами, AES маніпулює байтами. Він працює з даними у вигляді матриці розміром 4x4 байтів, яка називається станом (state).

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		29

Щоб змінити свій стан, алгоритм використовує SP-мережу (мережа замін-перестановок), з 10 раундами для 128-бітних ключів, 12 для 192-бітних ключів і 14 для 256-бітних ключів. Кожен раунд включає кілька основних операцій [2]:

- SubBytes: Кожен байт стану замінюється на інший байт за допомогою фіксованої таблиці підстановок (S-box);
- ShiftRows: Байти в кожному рядку матриці зсуваються на певну кількість позицій;
- MixColumns: Стовпці блоку перемішуються за допомогою лінійних перетворень;
- AddRoundKey: До кожного байту блоку додається відповідний байт раундового ключа за допомогою операції XOR.

Процес шифрування завершується після виконання всіх раундів, а останній раунд відрізняється від попередніх тим, що не включає крок MixColumns.

Безпека AES ґрунтується на складності підбору правильного ключа для дешифрування повідомлення без знання секретного ключа. Навіть з використанням найсучасніших комп'ютерів, спроба перебору всіх можливих ключів для AES-256 зайняла б мільярди років.

1.2.4 Асиметричні алгоритми шифрування

Концепція криптографії з відкритим ключем була вперше запропонована Вітфілдом Діффі, Мартіном Хелманом і Ральфом Мерклом. В асиметричних крипто-системах для шифрування використовується один ключ — відкритий (публічний), а для дешифрування — інший, закритий (приватний) [4].

Публічний і приватний ключі є великими числами, створеними на основі асиметричних алгоритмів. Відкритий ключ може бути доступний будь-кому, хто бере участь в обміні інформацією, проте знання публічного ключа не дозволяє обчислити відповідний закритий ключ.

Алгоритми, такі як RSA, ECC та DSA, використовують складні математичні обчислення для генерації ключів. Вони базуються на задачах, важких для розв'язання, таких як факторизація великих простих чисел або обчислення дискретних

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		30

логарифмів. Це робить їх практично неможливими для зламу, особливо при використанні довгих ключів.

Основна перевага асиметричної криптографії — можливість безпечного обміну ключами для подальшого використання у симетричних криптосистемах. Важливо забезпечувати надійний захист приватних ключів, оскільки їх втрата або компрометація може призвести до порушення конфіденційності даних.

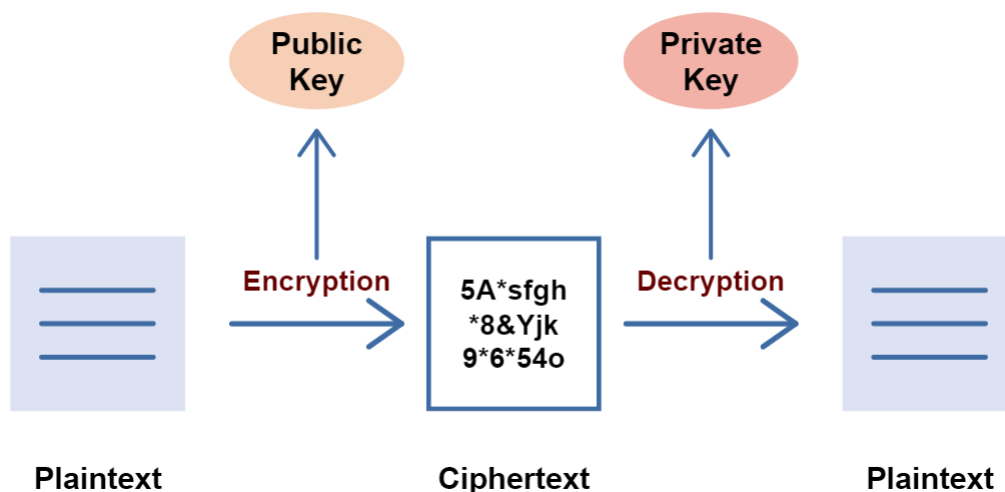


Рисунок 1.6 – Схема асиметричного шифрування

1.2.4.1 Rivest–Shamir–Adleman

Rivest-Shamir-Adleman (RSA) — це криптографічний алгоритм асиметричного шифрування, розроблений у 1977 році. Він використовується для шифрування та розшифрування даних і названий на честь своїх винахідників — Рональда Рівеста, Аді Шаміра та Леонарда Адлмана [2]. Безпека RSA базується на математичній складності факторизації великих чисел. Чим більші числа p і q , тим складніше факторизувати n й отримати приватний ключ d з публічного ключа e та n .

Основні етапи роботи RSA:

1) Генерація ключів:

- Вибір двох великих простих чисел p та q ;
- Обчислення їх добутку $n = p \times q$, який стає модулем для шифрування і розшифрування;
- Обчислення функції Ейлера $\phi(n) = (p-1) \times (q-1)$, яка визначає кількість додатних чисел, менших за n і взаємно простих з n ;

- Вибирається публічний ключ e , який задовольняє умову $1 < e < \phi(n)$ та $\gcd(e, \phi(n)) = 1$ (де $\gcd$ - найбільший спільний дільник);
- Обчислення числа d , яке є приватним ключем і оберненим до e за модулем $\phi(n)$, тобто $d \times e \equiv 1 \pmod{\phi(n)}$;
- Пара чисел (e, n) є відкритим ключем, а (d, n) — приватним ключем.

2) Шифрування:

- Повідомлення M перетворюється у ціле число m ($0 \leq m < n$);
- Обчислюється зашифроване повідомлення $C = M^e \pmod{n}$.

3) Розшифрування:

- Зашифроване повідомлення C перетворюється у число c ;
- Отримуємо дешифроване повідомлення $M = C^d \pmod{n}$.

RSA також використовується для створення цифрових підписів, що є важливою функцією в забезпеченні автентичності та цілісності інформації. Цифровий підпис дозволяє одержувачу перевірити, що повідомлення або документ були створені відправником і не були змінені після підписання.

Попри свою ефективність, RSA може бути вразливим до атак, зокрема до факторизації модуля n при використанні надто малих значень ключів. Тому для забезпечення високого рівня безпеки важливо обирати ключі RSA з достатньо великими значеннями.

1.2.5 Візуальна криптографія

Візуальна криптографія є специфічною галуззю криптографії, яка дозволяє шифрувати візуальну інформацію таким чином, щоб її можна було розшифрувати безпосередньо зоровою системою людини. Цей метод був вперше представлений в 1994 році Моні Наором та Аді Шаміром і відкрив нові можливості для захисту графічних даних. Основна ідея візуальної криптографії полягає у розбитті зображення на кілька частин, які окремо не дають жодної інформації, але при накладенні одна на одну відновлюють вихідне зображення [36].

Основні принципи візуальної криптографії:

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		32

- Розділення зображення: Початкове зображення розділяється на кілька шарів (зазвичай два або більше), кожен з яких окремо не несе інформації, а виглядає як випадковий набір пікселів;
- Піксельне перетворення: Кожен піксель початкового зображення перетворюється на кілька пікселів у кожному шарі. Ці пікселі можуть бути чорними або білими, залежно від початкового значення;
- Відновлення зображення: Для відновлення початкового зображення необхідно накласти всі частини одна на одну. Це може бути зроблено вручну шляхом накладання шарів зображення або за допомогою програмного забезпечення.

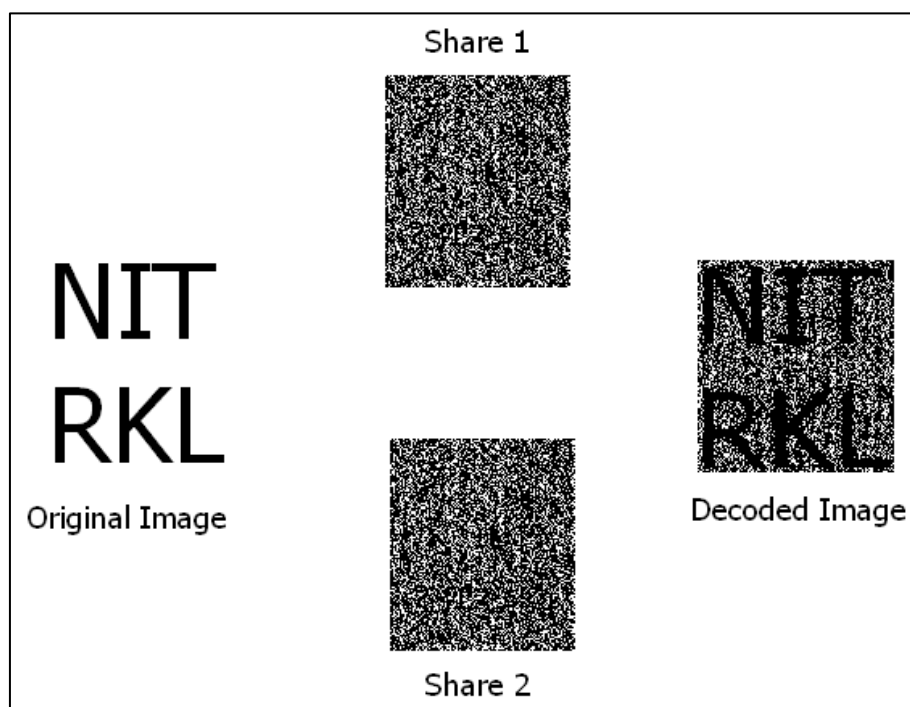


Рисунок 1.7 – Приклад використання візуальної криптографії

1.2.6 Кодування

American Standard Code for Information Interchange (ASCII) — це стандартний код для представлення тексту в комп'ютерах та інших пристроях, що використовують текст. ASCII було розроблено в 1960-х роках і залишається основоположним стандартом у комп'ютерних науках. Він використовує 7 біт для представлення кожного символу, що дозволяє закодувати 128 різних символів [14]. Ці символи включають:

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		33

- літери англійського алфавіту (великі та малі);
- цифри від 0 до 9;
- спеціальні символи, наприклад, пунктуаційні знаки, математичні символи);
- керуючі символи, наприклад, символи нового рядка, табуляції, зворотнього переносу).

Найбільшим обмеженням ASCII є його здатність кодувати лише англійський алфавіт та деякі спеціальні символи. Це стало очевидним з розвитком міжнародних комп'ютерних систем, що потребували підтримки багатьох різних мов та символів.

Unicode Transformation Format

Unicode Transformation Format (UTF) є стандартом для кодування символів Unicode, який включає літери та ієроєфі майже всіх письмових мов світу, а також технічні символи, знаки іконок тощо. Він був розроблений для усунення обмежень ASCII та забезпечення універсального стандарту для кодування тексту [14].

UTF-16 використовує 16 біт для представлення кожного символу. Це дозволяє закодувати символи з діапазону від 0x0000 до 0xFFFF. Однак для представлення символів поза цим діапазоном (так званих додаткових символів) UTF-16 використовує пари 16-бітових кодових одиниць, що називаються сурогатними парами. Це дозволяє UTF-16 кодувати до 1112064 різних символів. UTF-16 зазвичай використовується в середовищах, де компактність кодування та підтримка широкого діапазону символів є важливими, наприклад у Windows.

UTF-32 використовує 32 біти для представлення кожного символу. У цьому кодуванні кожен символ представлений одним 32-бітовим значенням, що робить його простим для інтерпретації. Однак, з точки зору пам'яті, це менш ефективно, оскільки для зберігання кожного символу потрібно більше байтів. UTF-32 може кодувати всі символи з діапазону Unicode, що робить його універсальним.

Base-кодування

Base-кодування — це методи перетворення даних в текстовий формат. Вони використовуються для передачі даних через середовища, які не підтримують бінарні дані, такі як електронна пошта або текстові документи.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		34

Base16, або шістнадцяткова система, використовує 16 символів (цифри від 0 до 9 і літери від A до F). Це дозволяє ефективно кодувати дані, використовуючи менше символів, ніж у десятковій системі. Кожен символ у Base16 представляє 4 біти інформації. Base16 широко використовується у комп'ютерних системах для представлення байтів у зручному для читання вигляді. Типовими прикладами є хеш-функції та MAC-адреси мережевих пристроїв. Також шістнадцятковий код застосовується у відображенні пам'яті комп'ютера, кольорів в веб-дизайні та інших технічних сферах.

Base32 використовує 32 символи, які включають великі літери латинського алфавіту (A-Z) та цифри від 2 до 7. Це зменшує ризик помилок при ручному введенні даних, оскільки виключаються схожі символи, такі як 1 і I, 0 і O. Кожен символ у Base32 представляє 5 біт інформації. Це робить Base32 компактнішим, ніж Base16. Це кодування часто використовується у системах, де потрібно передавати дані у текстовому форматі, зокрема у QR-кодах, певних криптографічних додатках та у випадках, коли необхідно уникнути нечітких символів.

Base64 використовує 64 символи, які включають великі та малі літери, цифри, а також два додаткові символи (+ та /). Цей набір символів забезпечує високу ефективність кодування даних. Кожен символ у Base64 представляє 6 біт інформації. Base64 широко використовується для кодування даних, які передаються через інтернет-протоколи [14].

1.2.7 Азбука Морзе

Азбука Морзе, або код Морзе, розроблена в 1830-х роках американським винахідником Самюелем Морзе та його асистентом Альфредом Вейлом. Вперше вона була використана для передачі повідомлень по телеграфу — новому на той час засобу зв'язку, який дозволяв «транспортувати» інформацію на великі відстані за допомогою електричних сигналів. Перше успішне повідомлення було передано 24 травня 1844 року, коли Морзе відправив слова «What hath God wrought» з Вашингтона до Балтимора.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		35

Код Морзе складається з коротких та довгих сигналів, відомих як точки і тире, які утворюють букви, цифри та розділові знаки. Кожен символ кодується певною комбінацією точок і тире. Наприклад: А: .-, Б: -..., В: ---, Г: --. Між сигналами одного символу робиться коротка пауза, між символами одного слова — середня пауза, а між словами — довга пауза.

Азбука Морзе часто використовується у криптографічних завданнях як один з етапів шифрування або як частина складнішого коду. Учасники повинні розпізнати і декодувати повідомлення, закодовані в Морзе, щоб отримати важливу інформацію чи ключ до розв'язання завдання. В СТФ-завданнях стеганографія може включати приховання повідомлень, закодованих азбукою Морзе, у зображеннях, аудіофайлах або інших медіа. Учасники повинні ідентифікувати і витягнути ці приховані повідомлення.

1.3 Стеганографічні методи та алгоритми для СТФ-змагань

Стеганографія – це метод приховування інформації в інших даних так, щоб її існування залишалось непоміченим. Наприклад, повідомлення можна сховати в тексті, зображенні, аудіо чи відео без зміни їх зовнішнього вигляду.

Історія стеганографії налічує тисячоліття. Потреба в захисті конфіденційної інформації завжди підштовхувала до приховування факту її існування, а розвиток різноманітних технічних, фізичних та психологічних методів надавав засоби для реалізації цього завдання. Початок стеганографії традиційно пов'язують із Єгиптом, але навіть наскельні малюнки стародавніх людей можна розглядати як своєрідні «стеганографічні повідомлення» [7].

Перша згадка про стеганографію в літературі відноситься до часів Давньої Греції. За словами Геродота, щоб передати повідомлення про плани перського царя Ксеркса щодо захоплення Греції, використовувалася звичайна дощечка з прихованим повідомленням під шаром воску.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		36

Часи Темного середньовіччя сприяли не лише появі інквізиції. Посилення стеження спричинило розвиток як криптографічних, так і стеганографічних методів. Саме в цей період вперше було застосовано комбінацію шифрування і приховування інформації за допомогою стеганографії [7].

З XVIII століття її активно використовували розвідники і революціонери, застосовуючи невидиме чорнило, дуже малі розміри повідомлень та інші витончені методи. Фотографічні мікрокрапки дозволили стискати розвідувальну інформацію до розміру крапки і вміщувати їх у простих листах. Це було популярно серед німецьких розвідників під час Другої світової війни [7].

Сучасна комп'ютерна стеганографія є важливою галуззю інформаційної безпеки, що досліджує методи приховування даних в обчислювальних системах і мережах. Існує кілька типів стеганографії: в тексті, зображеннях, аудіо, відео і навіть мережевому трафіку.

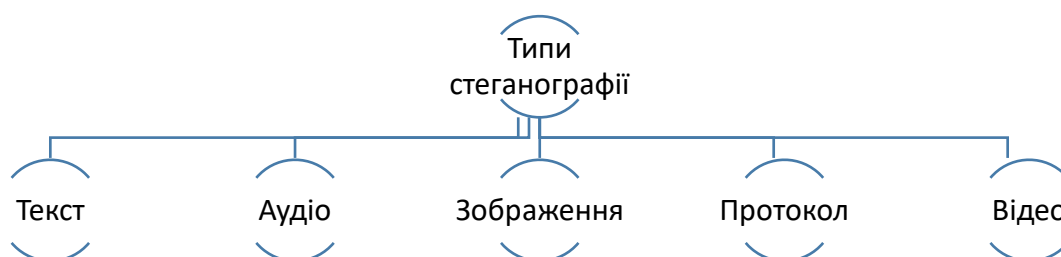


Рисунок 1.8 – Типи сучасної стеганографії

1.3.1 Стеганографія в тексті

Стеганографія тексту - це непростий процес приховання інформації всередині текстового повідомлення. Вона може бути складнішою за інші форми стеганографії, оскільки текстові файли мають менше «зайвої» інформації, яку можна використати для приховування даних. У текстових документах структура є досить стандартною і відомою, тому будь-які зміни можуть бути помічені навіть випадковим читачем. У той же час, зображення або аудіофайли мають складнішу структуру, що дозволяє ефективніше приховувати інформацію без помітних змін для людини.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		37

Стеганографія тексту може бути узагальнено класифікована на три типи:

- на основі формату,
- лінгвістичні методи,
- випадкова та статистична генерація.

Методи на основі формату використовують специфічні характеристики тексту, такі як орфографічні помилки, розмір шрифту та використання пробілів, для приховування інформації. Однак, цей підхід має певні недоліки. Якщо стегофайл відкрити за допомогою текстового редактора, орфографічні помилки та зайві пробіли легко виявляються. Змінений розмір шрифту може викликати підозру у читача. Крім того, якщо відомий оригінальний текст, його порівняння з модифікованим стеганографічним текстом може допомогти виявити змінені частини [8].

Лінгвістичні методи базуються на властивостях мови та тексту. Наприклад, використання певних слів, фраз або структур речень може служити як спосіб приховування інформації всередині тексту.

Методи випадкової та статистичної генерації застосовують певні алгоритми для створення тексту, який використовується для приховування інформації. Це може включати як генерацію тексту з випадкових символів, так і застосування статистичних моделей для вбудовування даних [8].

Основні підходи текстової стеганографії

1) Line Shift (Зсув рядка)

Метод «Line Shift» у стеганографії використовується для приховування секретних повідомлень шляхом вертикального зсуву рядків тексту. Основна ідея полягає в тому, що кожен рядок тексту може бути зсунутий вгору або вниз на певну величину, що кодує біти інформації. Наприклад, для біта 0 використовується зсув рядка вгору, а для біта 1 — зсув вниз.

Контрольні лінії допомагають визначити напрямок зсуву рядка. Кожен помічений рядок має дві непомічені контрольні лінії для визначення зсуву вгору чи вниз, що дозволяє правильно декодувати приховане повідомлення.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		38

Недоліки методу включають втрату інформації при переписуванні тексту або застосуванні програм для OCR. Спеціальні інструменти можуть фіксувати інтервали між рядками, які використовуються для зсуву, виявляючи приховані дані [8].

2) Word Shift (Зміщення слів)

Метод «Word Shift» приховує секретні повідомлення у тексті шляхом горизонтального зміщення слів вліво або вправо для кодування бітів 0 або 1. Кореляційний метод виявляє ці зсуви, аналізуючи профіль як хвилю і визначаючи, чи було слово зміщене вліво або вправо.

Перевагою цього методу є те, що зміщення слів може залишатися непомітним, оскільки зміна відстані між словами є звичним явищем. Однак недолік полягає в тому, що якщо хтось має доступ до алгоритму визначення відстаней між словами, він може розкрити приховане повідомлення, порівнюючи відкритий текст з стеганографічним.

3) White Steg (Метод пробілів)

Техніка «White Steg» у стеганографії використовує пробіли (прогалини, табуляції тощо), щоб приховати секретне повідомлення. У цій техніці є три основні методи:

- Інтервал між реченнями: один пробіл використовується для кодування біта 0, а два пробіли — для біта 1 після крапок або інших розділових знаків, що завершують речення;
- Інтервали між рядками: додається фіксована кількість пробілів у кінці кожного рядка. Наприклад, два пробіли кодують один біт, чотири пробіли — два біти і так далі;
- Інтервал між словами: один пробіл після слова означає біт 0, тоді як два пробіли представляють біт 1.

Один з недоліків техніки White Steg полягає в тому, що читачі можуть помітити непослідовне використання пробілів, і це може поставити під загрозу секретність прихованого повідомлення. Забезпечення послідовного та непомітного використання пробілів є вирішальним для ефективності цього методу.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		39

4) Feature Coding (Функціональне кодування)

Функціональне кодування — це стеганографічний метод, де секретне повідомлення приховується шляхом зміни певних характеристик тексту. Інструмент синтаксичного аналізу досліджує текст і визначає елементи, якими можна маніпулювати для приховування інформації. Наприклад, можна змінювати положення крапки в літерах, таких як «i» та «j», налаштовувати довжину штрихів у літерах, таких як «f» і «t», або змінювати висоту літер, як «b», «d», «h» тощо [8].

Недоліком цього методу є те, що якщо текст обробляється за допомогою програми оптичного розпізнавання символів чи набирається повторно, прихований вміст може бути втрачено або спотворено.

5) Syntactic Method (Синтаксичний метод)

Синтаксичний метод передбачає використання знаків пунктуації, таких як крапка, кома, тощо, для приховування бітів 0 і 1. Однією з проблем цього методу є необхідність точного визначення місць для вставки розділових знаків. Неправильне розміщення може бути помітним для читачів, що потенційно ставить під загрозу секретність прихованого повідомлення.

Таким чином, хоча синтаксичний метод може бути креативним способом приховування інформації, він вимагає ретельного розгляду та уваги до деталей, щоб гарантувати, що приховане повідомлення залишиться непоміченим.

6) Spam Text (Спам-текст)

Метод спам-тексту в стеганографії передбачає використання файлів HTML і XML для приховування фрагментів інформації. Цей метод включає кілька технік:

- Різні теги. Якщо використовуються різні початкові та закриваючі теги, це інтерпретується як біт 0. Якщо той самий тег використовується як для початку, так і для закриття (наприклад, <tag> замість <start> і </end>), це інтерпретується як біт 1;
- Пробіл у тегах. Біт 0 представлено відсутністю пробілу в тегу, а біт 1 — наявністю пробілу всередині тегу.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		40

Однак важливо переконатися, що ці зміни не порушують функціональність або зовнішній вигляд документа, оскільки це може викликати підозру та поставити під загрозу таємницю прихованого повідомлення.

7) SMS-Texting (SMS-повідомлення)

Мова SMS-повідомлень стосується скорочених слів, які зазвичай використовуються в SMS. У стеганографії ми можемо приховати двійкові дані, застосовуючи повну або скорочену форму слова. Цей процес передбачає створення кодової книги, яка поєднує слова з відповідними скороченими формами. Під час приховування інформації повна форма слова використовується для представлення біта 0, тоді як скорочена форма - для біта 1.

Однак ефективність цього методу залежить від узгодженості та розуміння кодової книги між відправником і одержувачем. Будь-які розбіжності або неправильне тлумачення скороченої форми слова можуть призвести до помилок у декодуванні прихованого повідомлення [8].

1.3.2 Стеганографія в зображеннях

Стеганографія в зображеннях - це один з методів захисту інформації, що полягає в прихованні даних всередині зображення. Цей процес дозволяє вбудовувати секретну інформацію у вигляді тексту, файлів або інших даних без видимих змін у самому зображенні.

Вона використовує слабкі місця зорової системи людини, включаючи зміни в кольорі, яскравості або розташуванні пікселів, які на перший погляд не впливають на візуальний вигляд зображення.

Аналіз форматів зображень

У змаганнях CTF, завдання часто включають роботу з різноманітними форматами зображень, що вимагає від учасників глибоких знань про їхні властивості та можливості. Різні формати зображень можуть використовуватися для стеганографії, аналізу зображень, та інших задач, пов'язаних із безпекою інформації.

Зображення можуть зберігатися у форматах, таких як BMP, JPEG, PNG, та GIF, кожен з яких має свої характеристики.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		41

1) BMP (Bitmap)

BMP-зображення не стискаються, що дозволяє зберігати кожен піксель без втрат. Це робить їх ідеальним вибором для стеганографії, оскільки прихована інформація не спотворюється при збереженні чи відкритті файлу. Структура BMP файлів є простою та зрозумілою, що полегшує процес вставки та вилучення таємного повідомлення. Через відсутність стиснення BMP файли можуть бути дуже великими, і це робить їх менш практичними для передачі або зберігання значної кількості зображень. Великий розмір файлів може зменшувати обсяг прихованої інформації, особливо при транспортуванні через мережу з обмеженою пропускнуою здатністю.

2) JPEG (Joint Photographic Experts Group)

JPEG є популярним форматом для стиснених зображень з втратами. Він забезпечує гнучкість у розмірах файлів та високу якість відтворення. Використання JPEG для стеганографії вимагає обережності, оскільки стиснення може призвести до втрати частини прихованої інформації. Проте можна застосовувати метод LSB для вставки даних у менш важливі біти зображення.

3) PNG (Portable Network Graphics)

PNG використовує стиснення без втрат, що дозволяє зберігати зображення у хорошій якості, і є ідеальним для стеганографії. Хоча PNG файли зазвичай менші за BMP, вони все ще можуть бути досить великими у порівнянні з JPEG. PNG-зображення часто використовуються для стеганографії, особливо через їхню здатність зберігати додаткову інформацію в альфа-каналі або в метаданих.

4) GIF (Graphics Interchange Format)

GIF підтримує анімацію, яка дозволяє приховувати дані у різних кадрах зображення, що може збільшити обсяг секретної інформації. Ці файли зазвичай мають невеликий розмір, і це робить їх зручними для передачі через Інтернет. Палітра обмежена до 256 кольорів, що знижує якість зображення та зменшує обсяг прихованої інформації.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		42

GIF має невеликий стеганографічний потенціал через свої обмежені можливості кольорів і відносно низьку якість. Однак, він все ще може бути використаний для приховування деякої інформації.

Магічні байти

Магічні байти (Magic Bytes), також відомі як магічні числа або сигнатури файлів, є особливими послідовностями байтів, які використовуються для ідентифікації форматів [26]. Ці послідовності зазвичай знаходяться в початковій частині файлів і використовуються різними програмами для визначення типу файлу, не зважаючи на його розширення або інші метадані.

Таблиця 1.5 – Магічні байти зображень

Формат зображення	Магічні байти
BMP (.bmp)	0x42 0x4D
JPEG (.jpg, .jpeg)	0xFF 0xD8 0xFF 0xE0
GIF (.gif)	0x47 0x49 0x46 0x38 0x39 0x61
PNG (.png)	0x89 0x50 0x4E 0x47 0x0D 0x0A 0x1A 0x0A
TIFF (.tiff, .tif)	0x49 0x49 0x2A 0x01

Огляд методів стеганографії в зображеннях

Методи стеганографії зображень різноманітні й мають свої унікальні підходи та застосування. Серед найбільш поширених методів виділяються такі [24]:

- зміна найменш значущих бітів (LSB) – метод полягає у зміні найменш значущих бітів пікселів зображення для вбудовування даних.
- розширений спектр – метод використовує техніки, схожі на ті, що застосовуються у системах бездротового зв'язку. Дані розподіляються по всьому зображенню з використанням псевдовипадкової послідовності.
- F5 – метод стеганографії, який використовує техніку DCT (дискретного косинусного перетворення) для вбудовування даних у JPEG-зображення.
- вбудовування палітри – цей метод використовується в зображеннях з обмеженою палітрою кольорів, наприклад, GIF. Вбудовування даних здійснюється шляхом зміни індексів кольорів у палітрі.
- маскування даних використовує методи, схожі на цифровий водяний знак.

Воно вбудовує дані в області зображення, що менш сприйнятливі до змін,

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

такі як тіні або області з високою текстурою. Це дозволяє зберігати дані навіть після стиснення або часткового пошкодження зображення.

Розглянемо методи, які часто зустрічаються на СТФ-змаганнях.

1) Метод заміни найменш значущих бітів

Метод найменш значущих бітів (LSB) є найбільш популярним у стеганографії завдяки своїй простоті та ефективності в роботі з зображеннями. Перш ніж зануритися в деталі цього методу, розглянемо основні компоненти цифрового зображення.

Цифрове зображення складається з матриці пікселів, кожен з яких має значення яскравості або кольору. Кольори пікселів визначаються трьома каналами: червоним (Red), зеленим (Green) і синім (Blue), відомими як RGB. Кожен канал може мати значення від 0 до 255, де 0 означає відсутність кольору, а 255 - максимальну інтенсивність. Наприклад, білий колір виходить при значенні 255 у всіх трьох каналах, а чорний — при 0. Різні комбінації значень каналів формують відтінки інших кольорів.

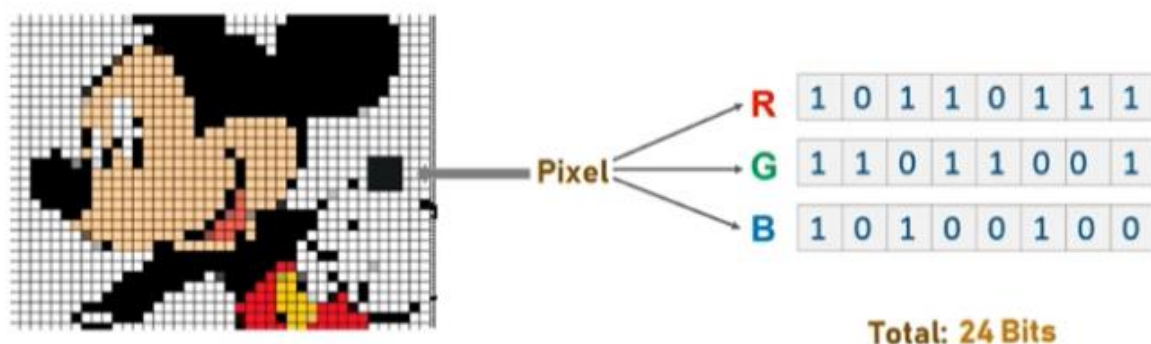


Рисунок 1.9 – Будова зображення

У методі LSB прихована інформація вбудовується у найменш значущі біти кожного каналу RGB. Це забезпечує мінімальні зміни в зображенні, які важко помітити неозброєним оком. Процес вбудовування інформації за допомогою методу LSB відбувається наступним чином:

- Обирається цифровий файл, наприклад, зображення, в яке буде вбудована інформація. Зазвичай використовуються формати, такі як BMP або PNG, оскільки вони не використовують стиснення з втратами;

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		44

- Інформація, яку потрібно вбудувати, перетворюється в двійковий формат. Наприклад, текстове повідомлення «Hello» подається у вигляді двійкового коду, який відповідає кожному символу в тексті;
- Повідомлення вбудовується в LSB пікселів послідовно. Наприклад, перший біт повідомлення записується в LSB червоного каналу першого пікселя, другий біт - в LSB зеленого каналу, і так далі. Після заповнення всіх LSB першого пікселя, процес переходить до наступного.

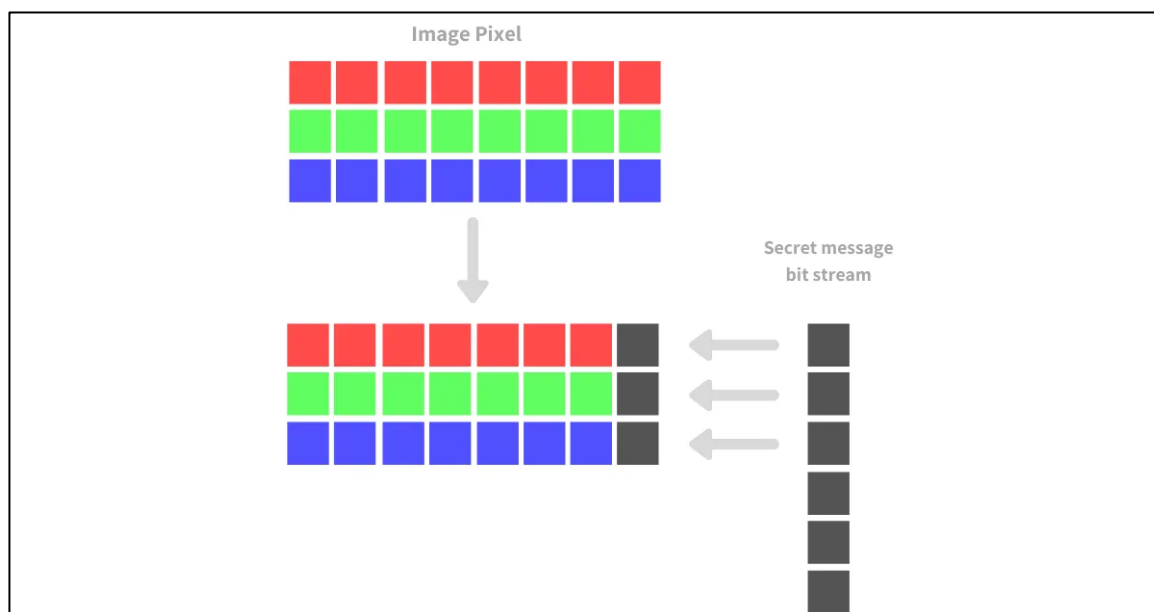


Рисунок 1.10 – Процес проховування даних у пікселях

Для вбудовування інформації в найменш значущі біти необхідно дотримуватися двох ключових принципів: збереження початкової візуальної якості зображення та мінімізація помітності внесених змін. Це означає, що зміни в LSB мають бути настільки незначними, щоб їх не можна було виявити візуально без використання спеціалізованих методів аналізу.

Зображення з прихованою інформацією зберігається у звичайному форматі. Для вилучення цієї інформації зображення обробляється шляхом зчитування найменш значущих бітів кожного пікселя. Потім ці біти об'єднуються у послідовність, щоб відновити початкове повідомлення.

2) Стеганографія на основі метаданих

Стеганографія на основі метаданих використовує додаткову інформацію, яка зберігається в файлах зображень, для приховання даних. Найбільш поширеним форматом для метаданих зображень є EXIF, який зазвичай застосовується у файлах JPEG та TIFF.

EXIF-дані — це додаткова інформація, що включає такі параметри, як дата і час зйомки, модель камери, параметри зйомки (витримка, діафрагма, ISO), GPS-координати, коментарі тощо. Ці дані можуть бути використані для приховання текстової або іншої інформації без зміни самого зображення.

Для приховування даних в метаданих зображень можна використовувати різні поля EXIF, такі як:

- User Comment: Поле для коментарів, яке може містити довільний текст.
- Artist: Поле для зазначення автора зображення.
- Copyright: Поле для інформації про авторські права.

Для роботи з EXIF-даними існує кілька інструментів, які дозволяють переглядати, редагувати і видаляти метадані з зображень.

3) Стеганографія у контейнерах

Стеганографія у контейнерах — це метод приховування даних, який використовує зображення як контейнер для зберігання інших файлів, таких як ZIP або RAR архіви.

Принцип роботи:

- Зображення використовується як контейнер для приховування файлів. Це можна зробити за допомогою об'єднання файлу зображення та архівного файлу, використовуючи командний рядок або спеціалізовані інструменти;
- Після об'єднання файлів вихідний файл все ще має вигляд зображення, але всередині містить прихований файл (архів);
- Для вилучення прихованих файлів учасники повинні ідентифікувати, що зображення містить додаткові дані. Це може бути зроблено за допомогою

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		46

аналізу структури та розміру файлу або використання спеціальних інструментів;

- Після виявлення прихованого файлу, учасникам варто вилучити його з контейнера. Це можна зробити за допомогою командного рядка та програм для роботи з архівами.

1.3.3 Стеганографія в аудіо

Аудіо стеганографія - це техніка прихованого передавання інформації у звукових сигналах. Вона передбачає вбудовування секретного тексту, зображень або інших даних безпосередньо у аудіофайли.

Аналіз форматів аудіофайлів

В сучасному світі існує безліч форматів аудіофайлів, кожен з яких має свої унікальні властивості та призначення. Аудіоформати можуть відрізнятися за типом стиснення, якістю звуку, розміром файлів та можливістю вбудовування додаткової інформації. Для завдань, де важлива висока якість звуку та збереження оригінальних даних, використовуються формати без стиснення, такі як WAV. Інші виклики можуть застосовувати популярні стиснуті формати, такі як MP3, для зменшення розміру файлів, хоча це може ускладнити процес стеганографії через втрату деяких даних під час стиснення.

1) MP3 (MPEG Audio Layer III)

MP3 використовує метод стиснення з втратами, що дозволяє значно зменшити розмір файлу, зберігаючи при цьому прийнятну якість звуку. Цей метод базується на психологічній акустиці, тобто видаляє звуки, які людське вухо не чує. Якість MP3-файлу залежить від бітрейту (бітова швидкість). Зазвичай, бітрейти коливаються від 128 кбіт/с до 320 кбіт/с. Вища бітова швидкість забезпечує кращу якість звуку, але й більший розмір файлу.

MP3 є одним з найбільш популярних аудіоформатів, сумісним з більшістю програмного та апаратного забезпечення. Підтримує вбудовування метаданих (теги ID3), які дозволяють зберігати інформацію про виконавця, альбом, рік випуску тощо.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		47

Завдяки широкому використанню та підтримці метаданих, MP3-файли можуть застосовуватися для приховання даних, хоча це ускладнюється через стиснення з втратами.

2) WAV (Waveform Audio File Format)

WAV файли зазвичай не стиснуті (PCM - Pulse Code Modulation), що означає, що вони зберігають всі оригінальні дані звукового сигналу, не втрачаючи якості. Існують також стиснені варіанти, але вони менш поширені. Цей формат забезпечує високу якість звуку, оскільки не застосовує стиснення з втратами. Часто використовується в професійному аудіовиробництві та звукозаписі.

Ці файли сумісні з більшістю аудіоплеєрів і програм для обробки звуку, хоча вони займають більше місця на диску через відсутність стиснення. Підтримують вбудовування метаданих, хоча й менш гнучко, ніж MP3.

WAV файли ідеально підходять для стеганографії через їх незжятий характер. В них легше вбудовувати приховані повідомлення, оскільки немає втрат інформації при обробці.

Основні методи аудіо стеганографії

1) Метод найменших значущих бітів (LSB)

Цей метод передбачає зміну найменш значущих бітів для приховання даних. Наприклад, якщо аудіо сигнал представлений у форматі з 16 бітами, то зміна найменших чотирьох бітів не суттєво вплине на якість сигналу. Приховані дані вставляються в ці біти, і для їх виявлення зазвичай використовується спеціальний ключ.

2) Методи широкосмугового кодування

Ці методи використовують модулювання сигналу псевдовипадковою послідовністю. Основна ідея полягає в тому, що одночасно змінюються декілька бітів у контейнері для приховування одного біта інформації. Наприклад, вихідний сигнал модулюється псевдовипадковою послідовністю $w(t)$ зі значеннями $\{-1, 1\}$. Результуючий стегосигнал $s(t)$ складається з вихідного сигналу $v(t)$ і прихованих даних $d(t)$ з коефіцієнтом загасання α : $s(t) = v(t) + \alpha * d(t) * w(t)$. Для витягання прихованих даних потрібна синхронізація псевдовипадкової послідовності $w(t)$ зі стегосигналом [7].

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				48
Змн.	Арк.	№ докум.	Підпис	Дата		

3) Метод приховування в луна-сигналі

Цей метод використовує властивість, що при невеликих зсувах у часі луна-сигнал практично не відчутний. Вставляються часові затримки ($\Delta 1$ для одиничного біта даних і $\Delta 0$ для нульового), що дозволяє внести відлуння в сигнал. Для відновлення даних застосовується автокореляційний аналіз, а стегоключ містить значення цих затримок [17].

4) Приховування файлів у контейнерах

Аудіо файли можуть використовуватись як контейнери для приховування інших файлів. Це реалізується шляхом вставки даних файлу в певні області аудіо сигналу або використанням методів, які дозволяють безпечно запаковувати файли в аудіо формат.

5) Приховування тексту в спектрограмі

Текст можна приховати в спектрограмі аудіо сигналу, змінюючи окремі частоти або амплітуди в частотному представленні сигналу. Наприклад, букви тексту можуть бути представлені у вигляді певних патернів частот, які незначно змінюються і не впливають на загальне звучання аудіо. Для виділення тексту необхідно провести спектральний аналіз сигналу.

1.3.4 Стеганографія у відео

Відео стеганографія – це методика приховування інформації у відеофайлах, яка дозволяє передавати секретні дані непомітно для сторонніх осіб. Вона ґрунтується на зміні певних характеристик відео таким чином, щоб вони були непомітними для людського ока, але могли бути легко відновлені за допомогою спеціальних алгоритмів.

Аналіз форматів відеофайлів

Існує безліч форматів відеофайлів, кожен з яких має свої унікальні властивості та призначення. Вибір формату може суттєво вплинути на ефективність стеганографічних методів.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		49

MP4 (MPEG-4 Part 14): Один з найпоширеніших відеоформатів, що забезпечує високий рівень стиснення без значної втрати якості. MP4 дозволяє вбудовувати додаткові потоки даних та метадані, що робить його популярним вибором для стеганографії.

AVI (Audio Video Interleave): Старіший формат, який має широку сумісність з різними програвачами та програмами. Однак AVI не підтримує сучасні методи стиснення та метадані, що може обмежити можливості стеганографії.

MOV (QuickTime File Format): Використовується здебільшого на платформах Apple. MOV-файли можуть містити різні медіа-потоки, що дозволяє вбудовувати приховані дані у відео.

FLV (Flash Video Format): Поширений у веб-відео, FLV використовує стиснення для забезпечення швидкого завантаження та відтворення. Однак стиснення з втратами може впливати на приховані дані.

Основні методи відео стеганографії

1) Метод найменш значущого біта (LSB)

Цей метод передбачає вбудовування даних шляхом зміни найменш значущих бітів пікселів у кадрах відео. Це мінімально впливає на якість відео, дозволяючи вбудовувати значну кількість даних. Прикладом може бути вбудовування текстових файлів у відео [35].

2) Просторово-часова стеганографія

Використання просторової та часової інформації (положення пікселів у кадрі та послідовність кадрів) для рівномірного розподілу даних по відео, що знижує ризик їх виявлення. Наприклад, дані можуть бути розподілені по різних частинах кадрів, щоб уникнути помітних змін.

3) Стеганографія на основі трансформацій

Приховування даних у частотній області відео за допомогою математичних перетворень, таких як дискретне косинусне перетворення (DCT) або дискретне хвильове перетворення (DWT), що забезпечує стійкість до компресії та інших обробок відео [35].

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		50

4) Методи на основі руху

Використання інформації про рух у відео для вбудовування даних, наприклад, зміни у векторних полях руху або інших параметрах, пов'язаних з переміщенням об'єктів у кадрі. Цей метод може бути ефективним для відео з динамічними сценами.

5) Метод кадрової різниці

Приховування даних шляхом аналізу різниці між послідовними кадрами відео. Цей метод ефективний для відео зі стабільним фоном і рухомими об'єктами. Наприклад, можна використовувати зміни у фоні для вбудовування даних.

6) Метод на основі тіньового кодування

Використання концепції розподілу секрету, де дані діляться на кілька частин і вбудовуються в різні кадри або відео. Відновлення даних можливе лише при доступі до певної кількості частин. Це підвищує рівень безпеки при передачі даних.

7) Використання метаданих

У деяких відеоформатах (наприклад, MP4, MKV) можна вбудовувати додаткові дані у метадані файлу. Це дозволяє зберігати приховану інформацію без внесення змін у самі відеокадри.

Висновки до першого розділу

Перший розділ присвячений детальному аналізу методів і засобів криптографії та стеганографії, які використовуються у CTF-змаганнях. Спочатку розглянуто організацію та проведення змагань у форматі CTF, де було підкреслено важливість таких подій для розвитку навичок у галузі кібербезпеки.

Основні поняття криптографії та класичні алгоритми шифрування формують фундаментальне розуміння основ шифрування. Симетричні алгоритми, зокрема Advanced Encryption Standard (AES), забезпечують високу швидкість та ефективність, тоді як асиметричні алгоритми, такі як Rivest–Shamir–Adleman (RSA), надають можливість безпечного обміну ключами. Важливе місце займає також візуальна криптографія та кодування, які часто використовуються для створення завдань у CTF-змаганнях.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		51

Методи стеганографії, розглянуті в цьому розділі, включають приховування інформації у текстах, зображеннях, аудіо та відео. Вони дозволяють учасникам змагань не лише шифрувати дані, але й приховувати сам факт їх існування, що надає додатковий рівень складності та інтересу до завдань.

Загалом, перший розділ заклав міцну теоретичну основу для подальшого дослідження та розробки практичних завдань у контексті CTF-змагань, зокрема з використанням криптографічних та стеганографічних методів.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		52

РОЗДІЛ 2 ПРОЕКТУВАННЯ ЗАВДАНЬ, АНАЛІЗ ТА ВИБІР ІНСТРУМЕНТІВ ДЛЯ ЇХ РЕАЛІЗАЦІЇ ТА РОЗВ’ЯЗАННЯ У CTF-ЗМАГАННЯХ

2.1 Методика розробки завдань для CTF-змагань

Методика – це система методів, прийомів і процедур, що використовуються для організації й проведення певного виду діяльності, зокрема, навчальної або дослідницької.

Ознайомившись із наявними підходами до розробки завдань, було вирішено створити свою методику. Ця методика враховує специфіку CTF-змагань та спрямована на забезпечення максимальної ефективності й цікавості викликів для учасників. Вона включає в себе системний підхід до планування, розробки, тестування та оцінки, що дозволяє створювати якісні й захоплюючі завдання.

Планування та розробка завдань для CTF-змагань - це творчий та захоплюючий процес, який вимагає уваги до дрібниць та креативності. Процес проектування завдання можна розбити на кілька етапів, які допоможуть систематизувати роботу та забезпечити успішне втілення ідей. Давайте розглянемо ці етапи докладніше.

Етап 1. Визначення мети та аудиторії

Визначення мети та аудиторії є важливим кроком при розробці завдань для CTF-змагань з кількох причин. Правильно визначена мета допомагає авторам чітко сформулювати задачу, підібрати відповідні підказки та ресурси, а також об'єктивно оцінювати рішення учасників. Знання цільової аудиторії, у свою чергу, дозволяє адаптувати складність завдань, вибрати відповідні терміни та інструменти, а також створити цікавий та доступний контент для учасників.

Етап 2. Вибір теми та концепції

Багато CTF-змагань мають певну тему, що допомагає привернути більше уваги учасників, які можуть виявити інтерес до конкретної тематики або області. Наприклад, змагання `osu!gaming CTF 2024` було створене розробниками комп'ютерної гри в жанрі музичних ритмів – «osu!». Всі завдання цього CTF були побудовані навколо різних фактів, пов'язаних з грою. Однак, брати участь у цьому змаганні було цікаво не лише фанатам, а й людям, які вперше дізналися про цю гру.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		53

Тематичні Capture The Flag змагання відкривають простір для креативності розробників завдань. Вони можуть використовувати знання теми, щоб створювати нешаблонні та неочікувані виклики, які стимулюють учасників мислити нестандартно та застосовувати різні підходи для їх вирішення.

Ми розглядали різні теми для CTF-змагань, такі як технології майбутнього, криптографія та стеганографія в стилі Шерлока Холмса або кібербезпека у світі магії та фентезі. Однак зупинилися на темі «Кібербезпека в мистецтві». Мистецтво є надзвичайно широким і різноманітним поняттям, що охоплює безліч форм і виражень. Від живопису та скульптури до музики, театру, літератури, мистецтво завжди відображало різні аспекти людського життя та досвіду. Кожна з цих областей може бути використана для передачі прихованих повідомлень або зашифрованих даних, що робить завдання цікавими та незвичайними для учасників.

Етап 3. Сюжетна лінія

Успішна сюжетна лінія у CTF змаганні подібна до тієї, що ви бачите у фільмах. Кожна частина завдання повинна мати логічне пояснення і поступово збільшувати інтерес гравців. Необхідно ретельно проробити сюжетну лінію, щоб вона логічно впліталася у виклики та заохочувала учасників. Також варто розглянути можливість додавання підказок у сюжет, щоб стимулювати учасників та підвищувати їхню зацікавленість.

Якщо ваше CTF-змагання має певну тематику, сюжетна лінія стане ідеальним місцем для використання та розвитку цієї теми, що дозволить отримати більше переваг в процесі вирішення завдань та допоможе поєднати їх між собою.

Етап 4. Складність та кількість балів

Складність завдань у CTF-змаганнях грає важливу роль у створенні захоплюючого та високоякісного досвіду для учасників. Завдання повинні мати різні рівні складності, щоб усі учасники могли знайти виклик, який відповідає їхнім знанням та навичкам. Від легких завдань для початківців до складних викликів для досвідчених учасників. Водночас важливо правильно оцінити кожне завдання у балах, щоб заохотити учасників до вирішення завдань різного рівня складності.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		54

Детальніше розглянемо можливі рівні складності завдань та відповідні їм бали:

– **Розминка (Warming up)**

Це завдання, які мають допомогти учасникам ознайомитися з СТФ, відпрацювати базові навички та набути впевненості перед тим, як братися за складніші виклики. Вони зазвичай дуже прості і призначені для того, щоб учасники могли зрозуміти основні механізми роботи платформи і формат подання відповідей. За такі завдання можна отримати від 5 до 50 балів.

– **Легкий рівень (Easy)**

Це завдання, які не потребують глибоких знань або досвіду в конкретній області. Їх можуть вирішити учасники, які тільки починають знайомитися з СТФ-змаганнями або не мають широкого досвіду в криптографії чи стеганографії. Наприклад, найпростіший шифр Цезаря чи різні символні шифри, кодування Base64 або перегляд метаданих зображення. Ці завдання оцінюються від 50 до 100 балів.

– **Середній рівень (Medium)**

Завдання вимагають від учасників певного рівня знань та досвіду. Вони охоплюють складніші аспекти криптографії та стеганографії, однак залишаються досяжними для тих, хто має базові навички в цих областях. Наприклад, виклики можуть стосуватися шифрування RSA або роботи з утилітою stegcracker. Учасники, які успішно вирішать ці завдання, отримують від 100 до 250 балів.

– **Складний рівень (Hard)**

Ці завдання розраховані на учасників з глибокими знаннями та значним досвідом у відповідних областях. Вони призначені для тих, хто готовий до складних викликів і прагне ґрунтовніше зрозуміти тему. Наприклад, завдання можуть включати використання непопулярних шифрів або спеціалізованих інструментів для стеганографії. Успішне виконання таких завдань приносить учасникам від 250 до 500 балів.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		55

– Надскладний рівень (Insane)

Завдання призначені для найкращих команд і вимагають вкрай глибоких знань, креативного мислення та досвіду. Вони можуть включати новітні або мало-відомі методи атак, потребувати розуміння складних протоколів або включати кілька етапів розв'язання. За вирішення таких завдань учасники можуть отримати від 500 до 1000 балів.

Варто зазначити, що розподіл балів для кожного CTF-змагання може відрізнатися в залежності від різних чинників, таких як платформа, тип змагання та структура виклику, а також умови, встановлені авторами завдань. Окрім того, кількість балів, яку можна отримати за вирішення кожного завдання, може змінюватися в ході змагання. Зокрема, зі збільшенням кількості розв'язків для певного виклику, його вартість у балах зменшується. Це спрощена стратегія для підтримки балансу та стимулювання учасників до розв'язання менш популярних або складних завдань. Такий підхід сприяє рівномірному розподілу зусиль між учасниками та робить змагання більш динамічним та захоплюючим.

Етап 5. Нашарування завдання

Нашарування завдань у CTF-змаганнях є необхідним елементом для створення захоплюючих та інтригуючих викликів. Цей підхід передбачає, що учасники, вирішивши одну частину завдання, отримують підказку або ключ для переходу до наступного етапу. Нашарування також включає розподіл завдання на кілька етапів, де кожен наступний етап стає доступним після успішного завершення попереднього. Це дозволяє створювати структуровані, логічно пов'язані виклики, які поступово розкриваються перед гравцями, забезпечуючи динамічний та інтерактивний досвід.

Етап 6. Елементи завдання

– **Назва завдання**

Створення яскравої та привабливої назви є важливим етапом у розробці завдання. Назва має не лише ідентифікувати виклик, але й зацікавити гравців, можливо, за допомогою підтексту або загадки. Такий підхід може збільшити мотивацію учасників знайти рішення. Наприклад, криптографічне завдання «Wheel Barrow» з

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		56

UofTCTF має назву, яка у оригінальний спосіб натякає гравцю на шифр, що використовувався для приховання прапора. Якщо здійснити пошук в Google «Wheel Barrow cipher online», ми знайдемо інформацію про перетворення Берроуза-Вілера, яке було використане для шифрування повідомлення. Стенографічне завдання «Strange-Historical-Machine» від SpringForwardCTF2024 теж містило натяк на всім відому шифрувальну машину часів Другої світової війни – Енігму.

– **Опис до завдання**

Опис є невід'ємною частиною створення цікавого та зрозумілого завдання у CTF-змаганнях. Перш за все, він розкриває суть завдання, пояснюючи учасникам, що саме вони повинні зробити. Наприклад, в описі можуть бути вказані певні кроки для розв'язання виклику, вимоги до використаних інструментів, формат прапора чи правила взаємодії з системою, якщо це потрібно.

Крім того, опис встановлює правила та обмеження, які учасники повинні дотримуватися під час розв'язання завдання. Це важливо, оскільки змагання має бути чесним і рівним для всіх учасників, тому необхідно чітко визначити, що дозволено та що заборонено під час вирішення завдання.

До того ж опис має бути досить креативним та різноманітним. Він може містити перифрази, загадки, жарти, цитати з відомих творів чи фільмів, що робить процес розв'язання завдання цікавішим та захоплюючим для учасників. Такий підхід не лише створює інтригу, але й стимулює творчість та аналітичні навички гравців. Наприклад, опис починається з загадкової фрази або літературної цитати, яка привертає увагу та заохочує учасників до роздумів. Також, він може містити неочікувані або цікаві факти, що додають привабливості завданню та спонукають учасників до досліджень.

Однак важливо не перевантажувати опис завдання різноманітністю, яка може приховати його суть або ускладнити розуміння учасниками. Чіткість та зрозумілість повинні залишатися пріоритетом навіть у креативних форматах опису.

– **Вкладені файли**

Кожне завдання, окрім оригінальної назви та цікавого опису, також має містити певні файли чи посилання на ресурси, які учасники мають досліджувати та

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		57

використовувати для розв’язку. Наприклад, це може бути текстовий файл з шифром для криптографічного завдання, зображення для стеганографічного виклику або інші дані, які потрібно обробляти чи аналізувати. Організатори можуть надати посилання на зовнішні ресурси, такі як веб-сайти, API, онлайн інструменти чи документацію, які містять додаткову інформацію або будуть використані для розв’язання завдання.

Етап 7. Формати прапорця

Прапор (англ. flag) — це унікальний текстовий рядок або значення, яке підтверджує успішне вирішення завдання. Прапор може бути різного формату, залежно від вимог організаторів та специфіки виклику.

Важливо, щоб прапор відображав суть завдання. Він повинен бути довгим і складним, схожим на парольну фразу довжиною більше ніж 15 символів, включаючи різноманітні комбінації літер, цифр і спеціальних символів.

Основні формати прапорів у CTF включають:

- Звичайний текстовий рядок, наприклад **CTF{G00d_J0b_Y0u_S0lved_It}**;
- Прапор, який був розділений на кілька частин і прихований в різних файлах. Наприклад, part1: «CTF{Nice_J0b}», part2: «_Y0u_G0t_», part3: «The_Flag}»;
- Прапорці можуть бути представлені як файли, які треба знайти в системі або отримати шляхом вирішення задач. Наприклад, **CTF{secret.txt}**;
- Деякі завдання можуть включати координати геолокації як прапорець, який вказує на місцезнаходження чогось в реальному світ. Наприклад, **CTF{50.15536, 28.4036}**;
- Прапорці можуть містити ідентифікатор уразливостей. Наприклад, **CTF{CVE-2024-1234}**.

У цих прикладах **CTF{...}** є початковим та заключним символами, які вказують на початок та кінець прапора, а між ними розташований унікальний текстовий рядок, який підтверджує успішне вирішення завдання.

Етап 8. Інструменти

Інструменти для CTF змагань можна поділити на дві категорії:

- інструменти для створення завдань;
- інструменти для їх розв'язання.

Нерідко ці дві категорії поєднуються між собою, адже однією консольною утилітою можна, наприклад, приховати дані, а далі їх виявляти. Для створення криптографічних завдань доволі часто використовуються мову програмування Python та різні додаткові бібліотеки.

Важливою частиною CTF-змагань є креативність у використанні інструментів та їх комбінацій для створення цікавих та складних завдань, які стимулюють учасників вирішувати виклики різними шляхами.

Етап 9. Проведення тестування

Розробку кожного завдання варто завершувати тестуванням, щоб перевірити функціональність та складність створених викликів. Наприклад, якщо це завдання з криптографії, варто перевірити, що шифротекст буде розшифрований правильно.

Крім того, тестування допомагає виявити та виправити можливі помилки та недоліки. Організатори можуть провести тестові запуски, щоб перевірити, як працюють системи під навантаженням та виявити будь-які проблеми або обмеження.

Варто бути впевненим, що завдання не має очевидних вразливостей, які будуть використані для обходу або спотворення правил. Ба більше, розробникам необхідно перевірити чи приховані дані в стеганографічних завданнях дійсно залишаються невидимими для звичайного користувача.

Етап 10. Документування процесу розробки завдань

Важливим аспектом є ведення документації щодо всіх етапів розробки завдань. Це включає опис концепцій завдань, деталі розробки, результати тестування та інструкції для учасників. Документація допоможе у майбутньому легше повторювати процес розробки та вдосконалювати завдання.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		59

2.2 Аналіз інструментів для розробки та вирішення криптографічних завдань

2.2.1 Онлайн ресурси та інструменти

Існує ціла низка онлайн ресурсів для створення шифротекстів та відновлення відкритого тексту. Деякі з цих інструментів спрощують процес шифрування та розшифрування для користувачів, надаючи їм доступ до різних криптографічних алгоритмів та методів. Інші можуть бути спеціалізовані на конкретних типах шифрування або аналізу шифротекстів.

CrypTool Online є одним з найпопулярніших онлайн інструментів для навчання криптографії. Цей ресурс надає інтерактивні модулі для роботи з різними криптографічними алгоритмами, такими як шифр Цезаря, Віженера, RSA та інші. CrypTool Online дозволяє користувачам шифрувати та дешифрувати повідомлення, аналізувати криптографічні протоколи та вивчати теорію криптографії через практичні завдання. Інтерфейс є зручним та інтуїтивно зрозумілим, що робить його корисним як для початківців, так і для досвідчених користувачів.

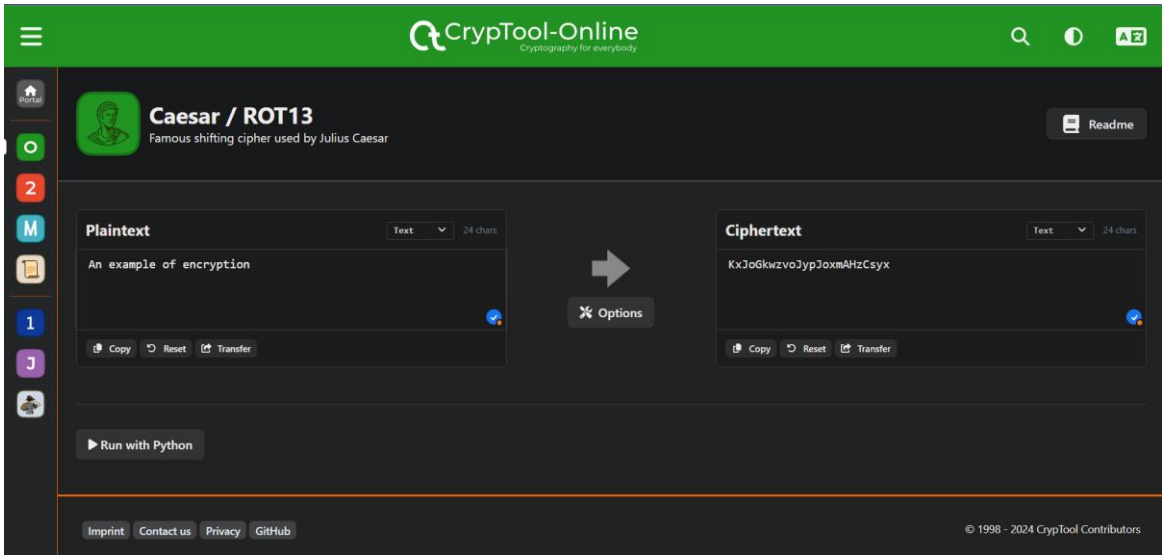


Рисунок 2.1 – Онлайн інструмент CrypTool

Сгуртії – це універсальний онлайн-інструмент для перетворення текстових даних за допомогою різних шифрів та кодувань. Він підтримує широкий спектр криптографічних алгоритмів та кодів, таких як Base64, Morse Code, Binary Code, та багато інших.

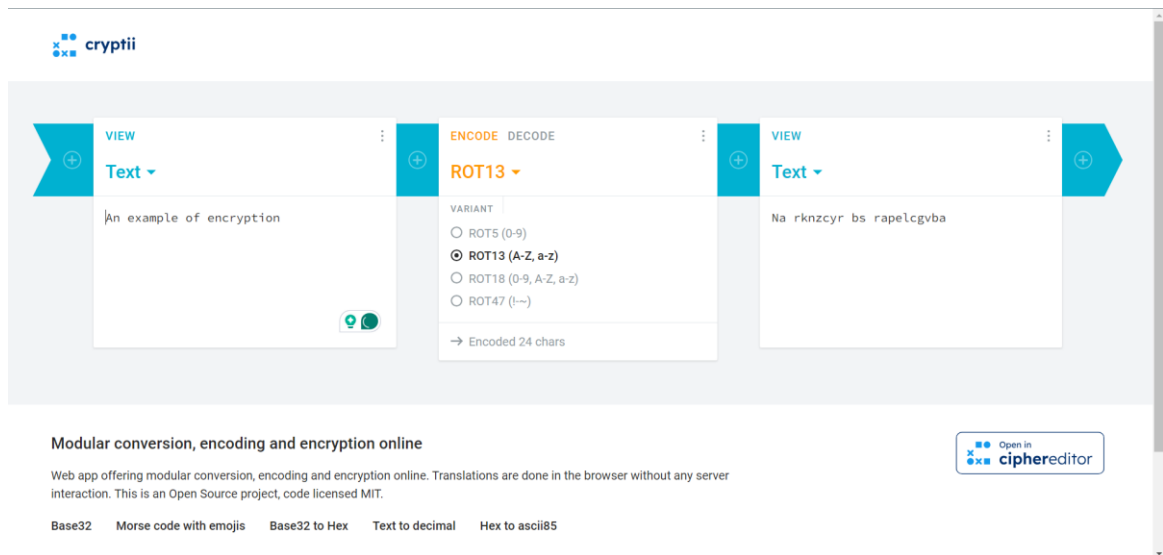


Рисунок 2.2 - Онлайн інструмент Cryptii

CyberChef, також відомий як «The Cyber Swiss Army Knife», є потужним інструментом для роботи з даними, який включає в себе численні функції для обробки, аналізу та трансформації даних. CyberChef підтримує широкий спектр криптографічних операцій, таких як шифрування, дешифрування, хешування, кодування та багато інших. Інтерфейс CyberChef дозволяє користувачам створювати складні сценарії обробки даних за допомогою простого перетягування блоків функцій, що робить його надзвичайно гнучким та зручним у використанні.

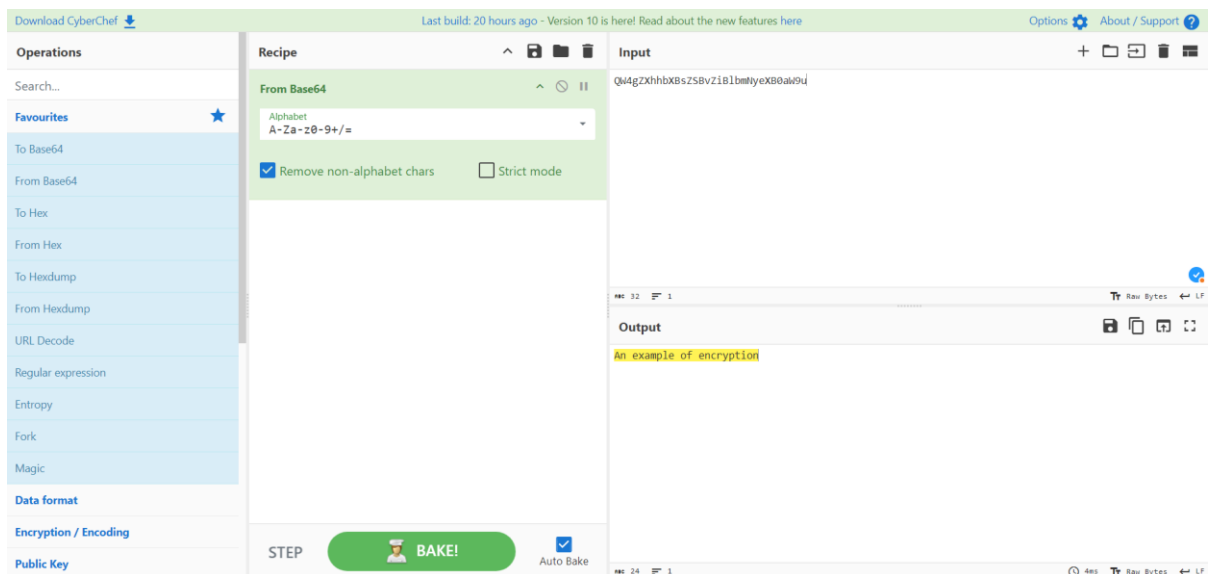


Рисунок 2.3 - Онлайн інструмент CyberChef

dCode - це онлайн-платформа, що надає великий набір інструментів для криптографії, стеганографії, математичного аналізу та інших задач. На dCode можна

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		61

знайти інструменти для роботи з класичними та сучасними шифрами, генерації паролів, розв'язання математичних рівнянь, аналізу тексту та багатьох інших завдань. Особливо корисним може бути Cipher Identifier, який допомагає ідентифікувати тип використаного шифрування або кодування.



Рисунок 2.4 - Онлайн інструмент dCode

2.2.2 Бібліотеки Python для криптографії

Python — це високорівнева, інтерпретована об'єктно-орієнтована мова програмування загального призначення, що набула широкої популярності завдяки своїй простоті вивчення та використання. Вона має лаконічний синтаксис, що робить її легкою для читання та розуміння, навіть для початківців. Код, написаний на Python, може запускатись на різних операційних системах без значних змін, що робить мову програмування універсальним інструментом для участі у CTF-змаганнях, де учасники використовують різні платформи.

Python має широкий вибір бібліотек, які спрощують вирішення різних завдань. Наприклад, Requests для взаємодії з веб-серверами та Pwntools для експлуатації вразливостей програмного забезпечення. Однак особливо важливі бібліотеки для криптографії, які дозволяють безпечно обробляти та захищати дані.

Нижче розглянемо декілька ключових бібліотек, які часто застосовуються у криптографічних завданнях:

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		62

1) **Crypto** - це набір бібліотек Python, призначений для роботи з криптографічними операціями. Він пропонує широкий спектр функцій для шифрування, дешифрування, аутентифікації та підпису цифрових даних [29].

Короткий огляд деяких модулів:

- **Crypto.Cipher**: цей модуль містить реалізацію алгоритмів шифрування із секретним ключем, таких як AES (Advanced Encryption Standard), DES (Data Encryption Standard) і ARC4. Він також підтримує шифрування з відкритим ключем за допомогою RSA PKCS#1;
- **Crypto.Hash**: модуль містить такі алгоритми хешування, як MD5, SHA (включаючи SHA-1, SHA-224, SHA-256, SHA-384 і SHA-512) і HMAC (Keyed-Hashing for Message Authentication Code). Хеш-функції необхідні для створення унікальних «підписів» даних;
- **Crypto.PublicKey**: тут ви знайдете реалізацію алгоритмів шифрування з відкритим ключем, таких як RSA (Rivest-Shamir-Adleman), і алгоритмів цифрового підпису, таких як DSA (Digital Signature Algorithm);
- **Crypto.Util**: пропонує різноманітні службові функції та модулі, включаючи перетворення, генерація випадкових чисел і теоретико-числові функції, які використовуються в криптографічних операціях.

2) **NumPy** відкриває безліч можливостей для роботи з числовими даними в Python. Однією з її ключових особливостей є використання масивів, що дозволяє зручно та ефективно зберігати та обробляти великі об'єми даних. NumPy також забезпечує широкий набір математичних функцій, завдяки яким можна здійснювати різноманітні обчислення, від простих арифметичних операцій до складних алгоритмів лінійної алгебри. Ця бібліотека корисна в криптографії для виконання матричних операцій та інших числових обчислень [28].

3) **Gmpy2** - це бібліотека, призначена для роботи з великими числами. Вона дозволяє виконувати різноманітні математичні операції з цифрами, які перевищують межі стандартних числових типів Python. Бібліотека має розширений набір математичних функцій, включаючи факторизацію, обчислення

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		63

квадратних коренів, визначення простоти числа і багато інших. Ще однією перевагою `gmpy2` є швидкість і ефективність виконання операцій з числами. Вона використовує оптимізовані алгоритми, що дозволяє виконувати складні математичні обчислення з високою точністю за короткий час [30].

2.2.3 RSA інструменти

RSA є одним з найбільш поширених алгоритмів асиметричної криптографії, який використовується для захисту даних. Існує ряд інструментів, що дозволяють працювати з RSA-ключами та реалізовувати різні види атак для їх відновлення. Розглянемо деякі з них:

1) RSACTFTool

RSACTFTool — це інструмент, призначений для відновлення RSA-приватного ключа за допомогою різних атак. Він є корисним для кібербезпеки, оскільки дозволяє фахівцям тестувати міцність RSA-криптографії та виявляти потенційні вразливості [25]. Основні можливості RSACTFTool включають:

- Використання різних атак. Інструмент підтримує різні типи атак, такі як атака на невеликі значення d , атака на частково відомий ключ та інші;
- Автоматизація процесу. Інструмент автоматично визначає найбільш підходящу атаку на основі наданих параметрів;
- Зручний інтерфейс. Командний рядок дозволяє швидко і ефективно використовувати інструмент;

2) Factordb

FactorDB — це база даних простих чисел та їх множників. Вона дозволяє знаходити фактори заданих чисел, що може бути корисним для аналізу криптографічних ключів. Цей ресурс стане в пригоді для дослідників та ентузіастів, які займаються факторизацією великих чисел у рамках криптоаналізу RSA. FactorDB також надає API для автоматизації процесу пошуку факторів, що може бути інтегровано в інші інструменти для ефективнішого аналізу.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		64

3) Alpertron ECM

Alpertron ECM (Elliptic Curve Method) — це онлайн-інструмент для факторизації великих чисел з використанням методу еліптичних кривих. Цей метод є одним з найефективніших для розкладання чисел на прості множники, що може бути використано для атаки на RSA ключі, якщо модуль (n) не був обраний належним чином. Alpertron ECM дозволяє виконувати факторизацію без необхідності встановлення додаткового програмного забезпечення, що робить його зручним для швидких тестів та навчальних цілей.

4) SageMath

SageMath (або просто Sage) — це потужна математична платформа, яка включає широкий набір інструментів для криптоаналізу. Зокрема, вона підтримує функції для факторизації чисел, генерації RSA ключів або атаки на слабкі ключі та інші криптографічні операції.

2.3 Аналіз інструментів для розробки та вирішення стеганографічних завдань

2.3.1 Інструменти для зображень

1) Графічні редактори

Adobe Photoshop є одним з найпопулярніших графічних редакторів у світі, що широко використовується для створення та редагування зображень. Цей редактор надає широкий спектр можливостей для роботи з різними форматами зображень, включаючи редагування кольорових каналів та пікселів. У контексті стеганографії, Photoshop може використовуватися для вбудовування прихованих даних в зображення за допомогою різних методів, таких як Least Significant Bit (LSB) метод. Photoshop забезпечує високий рівень точності та контроль над зображеннями, що робить його корисним інструментом для стеганографії.

GIMP (GNU Image Manipulation Program) - це потужний безкоштовний графічний редактор, що за своїми можливостями не поступається Photoshop. Він має широкий функціонал для редагування та маніпулювання зображеннями, включаючи

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		65

ючи роботу з векторною графікою, ретушування, маскування та створення спеціальних ефектів. GIMP підтримує різні формати зображень та може використовуватися для стеганографії, зокрема для вбудовування даних у LSB. Завдяки відкритому вихідному коду, користувачі GIMP можуть змінювати і доповнювати його функціонал для створення власних методів стеганографії.

2) Інструменти для вбудовування даних

Steghide - це програмне забезпечення для приховування даних у різних типах файлів, включаючи зображення та аудіофайли. Steghide підтримує формати BMP та JPEG для зображень. Інструмент дозволяє користувачам вбудовувати та витягувати приховані дані, забезпечуючи при цьому захист паролем для додаткової безпеки [31].

OpenStego - це безкоштовне програмне забезпечення для стеганографії, яке підтримує формати BMP та PNG. OpenStego дозволяє користувачам приховувати дані у зображеннях та забезпечує додаткові функції, такі як шифрування прихованих даних для підвищення безпеки.

Outguess - це інструмент для стеганографії, який підтримує формати JPEG, PPM та PNM. Він дозволяє приховувати дані у зображеннях та забезпечує функції для вбудовування та вилучення прихованої інформації.

3) Інструменти для аналізу та виявлення стеганографії

Pngcheck - це інструмент для перевірки цілісності та вмісту PNG-файлів. Він може бути використаний для виявлення прихованих даних та аномалій у PNG-зображеннях [31].

Zsteg - це утиліта для аналізу стеганографічного вмісту у PNG- та BMP-файлах. Вона здатна виявляти різні методи приховування даних, такі як LSB-стеганографія, і забезпечує детальний аналіз файлів.

StegOnline - це веб-додаток, який дозволяє користувачам виконувати стеганографічні операції без необхідності встановлення програмного забезпечення. StegOnline підтримує формати PNG та JPEG, забезпечуючи простий інтерфейс для вбудовування та вилучення прихованих даних.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		66

Aperisolve - це онлайн-сервіс для автоматичного аналізу та виявлення стеганографічного вмісту у файлах зображень. Цей інструмент підтримує різні формати зображень, такі як PNG, JPEG, BMP, GIF, та інші. Aperisolve виконує різноманітні методи аналізу, включаючи пошук прихованих даних у кольорових каналах, перевірку метаданих, а також виявлення файлів та повідомлень. Він забезпечує зручний веб-інтерфейс, що дозволяє користувачам легко завантажувати зображення та отримувати детальні результати аналізу.

Stegsolve - це утиліта для аналізу стеганографічного вмісту у зображеннях. Вона надає різноманітні інструменти для візуального аналізу зображень, включаючи перегляд окремих кольорових каналів, фільтрацію та інші методи.

4) Інструменти для ручного аналізу

HexEd.it - це онлайн-редактор двійкових файлів, який дозволяє редагувати та аналізувати вміст файлів на рівні байтів. Він може бути використаний для ручного аналізу та маніпулювання файлами, включаючи виявлення та зміну стеганографічних даних.

2.3.2 Інструменти для аудіо та відео

1) Аудіоредактори та аналізатори

Audacity - це безкоштовний, відкритий і багатоплатформовий аудіоредактор та рекордер. Він надає можливості для запису звуку, редагування аудіофайлів, накладання ефектів і збереження результатів у різних форматах. Audacity підтримує численні плагіни, що дозволяє значно розширити його функціональність. У стеганографії Audacity можна використовувати для аналізу та маніпулювання аудіофайлами з метою приховування або виявлення повідомлень.

SonicVisualizer - це програма для аналізу аудіо записів. Вона дозволяє візуалізувати та вивчати аудіофайли за допомогою спектрограм, хвильових форм та інших методів візуалізації. SonicVisualizer корисний для стеганографії, оскільки надає зручні інструменти для детального аналізу спектрального складу аудіосигналів, що допомагає у виявленні прихованої інформації.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		67

2) Інструменти для відеоредагування

Microsoft Clipchamp - це веб-застосунок для відеоредагування, що дозволяє створювати і редагувати відео за допомогою різноманітних інструментів та ефектів. Clipchamp підтримує імпорт та експорт різних відеофайлів, а також пропонує можливості для стискання відео та конвертації форматів. У контексті стеганографії Clipchamp може бути корисним для приховування повідомлень всередині відеофайлів шляхом редагування кадрів або використання спеціальних відеоефектів.

3) Інструменти для приховування та декодування

Deepsound – це інструмент для стеганографії, який дозволяє вбудовувати приховані повідомлення у звукові файли. Він підтримує різні формати аудіо файлів, такі як WAV та FLAC. Deepsound дозволяє шифрувати приховані повідомлення, що робить їх недоступними для несанкціонованого доступу [31].

mp3stego – це програма для приховування даних у MP3 файлах. Вона використовує стеганографічні методи для вбудовування текстових повідомлень у звукові файли без значного впливу на якість звуку. mp3stego забезпечує простий інтерфейс для шифрування та дешифрування прихованих даних.

QSSTV – це програма для прийому та передачі зображень за допомогою SSTV через аматорське радіо. QSSTV підтримує різні моди SSTV, такі як Robot, Scottie, та Martin. Програма дозволяє декодувати зображення, передані через аудіо сигнали, та може бути використана для стеганографії, де картинки містять приховані повідомлення. QSSTV має зручний інтерфейс та додаткові функції для налаштування параметрів прийому та передачі сигналів [37].

SSTV Decoder – це інструмент для декодування SSTV сигналів, які передаються через аудіо канали. Наприклад, програмне забезпечення з відкритим кодом на [GitHub](#), дозволяє декодувати зображення, передані за допомогою SSTV [38].

2.3.3 Інструменти для текстових файлів

1) Редактори текстових файлів

Notepad++ — це безкоштовний текстовий редактор з відкритим вихідним кодом, який підтримує безліч мов програмування. Він широко використовується для

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		68

редагування текстових файлів завдяки своїм розширеним можливостям, таким як підсвічування синтаксису, розширені функції пошуку та заміни, підтримка макросів і плагінів. Notepad++ можна використовувати для редагування текстових файлів, в яких приховано стеганографічну інформацію, а також для аналізу структури таких файлів.

2) Інструменти для стеганографії

Stegsnow — це стеганографічний інструмент, який використовує пробіли та табуляції для приховування інформації в текстових файлах. Інструмент був розроблений для роботи з текстовими файлами, забезпечуючи можливість приховування даних без зміни візуального вигляду тексту. Використання Stegsnow передбачає додавання зайвих пробілів та табуляцій у кінці рядків, які потім можуть бути декодовані для отримання прихованої інформації.

Unicode Steganography — це онлайн-інструмент для приховування тексту всередині інших текстів за допомогою спеціальних символів Unicode. Цей інструмент використовує різні варіації символів, що виглядають однаково, але мають різні коди Unicode, щоб приховувати повідомлення. Unicode Steganography дозволяє приховувати інформацію у звичайних текстових файлах, не змінюючи їх видимий зміст.

3) Інструменти для злому стеганографії

Snowcracker — це інструмент для зламу стеганографії Stegsnow. Він призначений для автоматичного аналізу текстових файлів з метою виявлення та розкодування прихованої інформації, схованої за допомогою пробілів і табуляцій [42].

Stegsnowbruteforcer — це ще один інструмент для зламу Stegsnow, який використовує метод брутфорс для знаходження прихованої інформації. Він автоматично перебирає різні можливі варіанти розташування пробілів та табуляцій у текстовому файлі, щоб відновити приховану інформацію [41].

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		69

2.4 Базові інструменти для CTF-завдань

2.4.1 Інструменти для початкового аналізу

У CTF-змаганнях базові інструменти для початкового аналізу є важливою частиною процесу розв'язання завдань. Вони допомагають отримати загальне уявлення про тип файлу, його структуру та можливі методи його подальшого аналізу.

Нижче наведені деякі з основних інструментів, які використовуються для початкового аналізу.

Команда **file** - це утиліта в UNIX-подібних операційних системах, яка призначена для визначення типу файлу шляхом аналізу його вмісту. Вона допомагає користувачеві ідентифікувати формат та властивості файлів навіть у випадку, якщо їх розширення не вказано або зазначено неправильно [32].

Команда **strings** використовується для пошуку та виведення послідовностей друкованих символів у файлах. Це корисно для виявлення прихованих текстових даних або підказок, що можуть міститися всередині двійкових файлів.

ExifTool - це потужна утиліта командного рядка для аналізу, редагування та витягування метаданих з різних типів файлів, особливо зображень. Вона може відображати детальну інформацію про метадані, такі як EXIF, IPTC, XMP та інші, що містяться в файлі [32].

Основні можливості ExifTool:

- Відображає детальну інформацію про метадані, такі як параметри зйомки, авторська інформація, географічні дані;
- Дозволяє додавати або змінювати ключові слова, опис, авторську інформацію, дату та час зйомки;
- Можливість витягувати метадані з файлів і зберігати їх у текстовому або іншому форматі для подальшого аналізу чи обробки;
- Підтримує багато різних форматів файлів, включаючи JPEG, TIFF, RAW, PNG, PDF, AVI, MP4 та інші.

Окрім цієї утиліти можна використовувати, наприклад, онлайн інструмент **metadata2go**, який має зручний дизайн, що дозволяє швидко та ефективно працювати з метаданими файлів.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		70

Foremost - це безкоштовна утиліта командного рядка для відновлення файлів з пошкоджених або нерозпізнаних носіїв даних. Вона використовує різні алгоритми для пошуку та вилучення файлів на основі їх сигнатур (унікальних патернів байтів), що робить її ефективним інструментом для дослідження даних і пошуку прихованих файлів[33].

Binwalk - це потужний інструмент для аналізу та вилучення вбудованих файлів і даних з двійкових образів. Він особливо корисний для аналізу вбудованого програмного забезпечення, мікропрограм та інших типів двійкових файлів. Binwalk дозволяє ідентифікувати і витягувати різні компоненти з файлів, такі як архіви, зображення, виконувані файли та багато іншого.

2.4.2 Інструменти для роботи з архівами

Багато CTF-завдань містяться в архівах типу zip, 7z, rar, tar або tgz. Проте, коли сам архівний файл є частиною виклику, учасникам варто використовувати різні інструменти для роботи з архівами, особливо коли вони пошкоджені або захищені паролями. Нижче наведено основні інструменти та їх можливості для роботи з архівами.

1) Основні інструменти для роботи з архівами

Unzip: Це базовий інструмент для розпакування ZIP-файлів. Він підтримує різні опції, що дозволяють не тільки розпаковувати архіви, але й отримувати додаткову інформацію про їх вміст, змінювати спосіб розархівування та виконувати інші конкретні дії.

WinRAR: Популярний інструмент, відомий своєю здатністю працювати з RAR-архівами, але він також підтримує широкий спектр інших форматів, таких як ZIP, 7z, TAR. Завдяки своїй універсальності та потужним можливостям стиснення, WinRAR є улюбленим вибором багатьох користувачів для створення та розпакування архівів.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		71

7-Zip: Потужний архіватор, який підтримує численні формати архівів, включаючи ZIP, RAR, 7z, TAR, GZIP та інші. 7-Zip відомий своїм високим рівнем стиснення та відкритим вихідним кодом, що робить його доступним для багатьох користувачів безкоштовно.

2) Інструменти для детального аналізу та відновлення архівів

Zipdetails: Утиліта для аналізу та перегляду вмісту ZIP-файлів, надає детальну інформацію про кожен файл і його властивості.

Zipinfo: Утиліта, що входить до складу пакету Info-ZIP і використовується для отримання детальної інформації про вміст ZIP-архівів. Вона дозволяє переглядати структуру архіву, включаючи файли і директорії, а також отримувати інформацію про кожен файл у архіві, таку як розмір, дата створення і зміни, рівень стиснення та інші метадані.

zip -F та zip -FF: Команди для відновлення пошкоджених ZIP-файлів. Перша виконує базове відновлення файлу, тоді як друга використовує додаткові методи для відновлення сильно пошкоджених файлів. Ці команди можуть бути надзвичайно корисними для збереження важливої інформації у випадку пошкодження архіву.

3) Інструменти для роботи з захищеними паролем архівами

fcrackzip: Спеціалізований інструмент, призначений для брутфорс-атак на ZIP-файли, які захищені паролем. Він дозволяє автоматизувати процес підбору пароля до архіву, що є дуже корисним при роботі з зашифрованими файлами [39].

John The Ripper: Потужний інструмент для зламу паролів, який також може використовуватись для роботи з зашифрованими архівами. Він підтримує різні методи атак, включаючи словникові атаки та брутфорс-атаки, що робить його універсальним засобом для розшифровки захищених файлів. John The Ripper відомий своєю здатністю ефективно знаходити слабкі місця у захисті паролями та має багатий набір опцій для налаштування атак [40].

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		72

Висновки до другого розділу

У другому розділі було детально розглянуто процес проектування завдань для CTF-змагань, а також аналіз та вибір інструментів для їх розробки та вирішення. Основну увагу приділено методиці створення завдань і критеріям вибору відповідних інструментів для різних типів викликів.

Окремо висвітлено криптографічні засоби, зокрема онлайн-ресурси, бібліотеки для Python і RSA інструменти. Вони дають змогу створювати завдання різної складності та забезпечувати їх правильне розв'язання, що є критично важливим для підтримання високого рівня змагань.

Також досліджено стеганографічні інструменти для зображень, аудіо, відео та текстових файлів. Використання стеганографії дозволяє приховувати інформацію у медіафайлах, додаючи додатковий рівень складності для учасників.

Крім того, представлено базові інструменти для початкового аналізу та роботи з архівами. Вони допомагають учасникам ефективно розпаковувати архіви і проводити первинний аналіз файлів.

Таким чином, другий розділ надає всебічний огляд інструментів та методів, які використовуються для створення і виконання CTF-завдань, формуючи повне розуміння процесу їх розробки та реалізації.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		73

РОЗДІЛ 3 ПРАКТИЧНА РЕАЛІЗАЦІЯ КРИПТОГРАФІЧНИХ ТА СТЕГАНОГРАФІЧНИХ ЗАВДАНЬ ДЛЯ STF-ЗМАГАНЬ

3.1 Практична реалізація криптографічних завдань легкого рівня

3.1.1 «Julius Caesar»

Мета реалізації:

Дане завдання зосереджене здебільшого на застосуванні одного із класичних шифрів, однак воно також містить елементи стеганографії зображення.

Завдання спрямовуване на розвиток аналітичних навичок учасників, оскільки їм потрібно не лише розгадати зашифроване повідомлення, а спершу виявити його у зображенні.

Використані техніки: шифр Цезаря, стеганографія зображення

Мови програмування/інструменти: мова програмування Python, HexEd.it

Вхідні дані:

– *Назва завдання:* «Julius Caesar»

– *Опис до завдання:*

Серед величних та могутніх імператорів минулого стоїть Юлій Цезар, фігура, що викликає захоплення та інтригу. Він - символ влади, військового мистецтва та політичного впливу, чиє ім'я звучить як вічне свідчення про величні подвиги.

Спробуйте розгадати, яку таємницю містить його скульптура.

– *Формат прапора:* після виконання завдання ви отримаєте прапор формату flag{Слово1_Слово2}

– *Файли для розв'язання завдання:* зображення Caesar Cipher.jpg

– *Підказки:*

Підказка №1. У книзі, яку я нещодавно прочитав, головний герой завжди використовував кодове число для доступу до схованого скарбу. Це число, як не дивно, було 19.

Підказка №2. Спробуйте нестандартний підхід у виборі алфавіту.

Алгоритм/процес реалізації:

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		74

1) Придумуємо прапорець, які відповідає темі завдання. Основою для прапора може бути один із крилатих виразів римського імператора, скажімо: «I came, I saw, I conquered».

Наприклад, можливі варіанти:

- flag{1_C4m3_1_S4w_1_C0nqu3r3d};
- flag{V3n1_V1d1_V1c1};
- flag{3xper13nce_1s_7h3_7each3r_0f_4ll_7h1ngs}.

2) Для шифрування прапора пишемо код. Як ключ обираємо випадкове число 19.

Прапор після шифрування: y4tz{k_VN5m_k_BNf_k_VJ69dmamw}

3) Для того, щоб приховати прапор у змісті картинки скористаємось шістнадцятковим онлайн-редактором. У hex полі знаходимо нульові байти та замінимо їх секретним повідомленням. Зберігаємо змінену картинку.

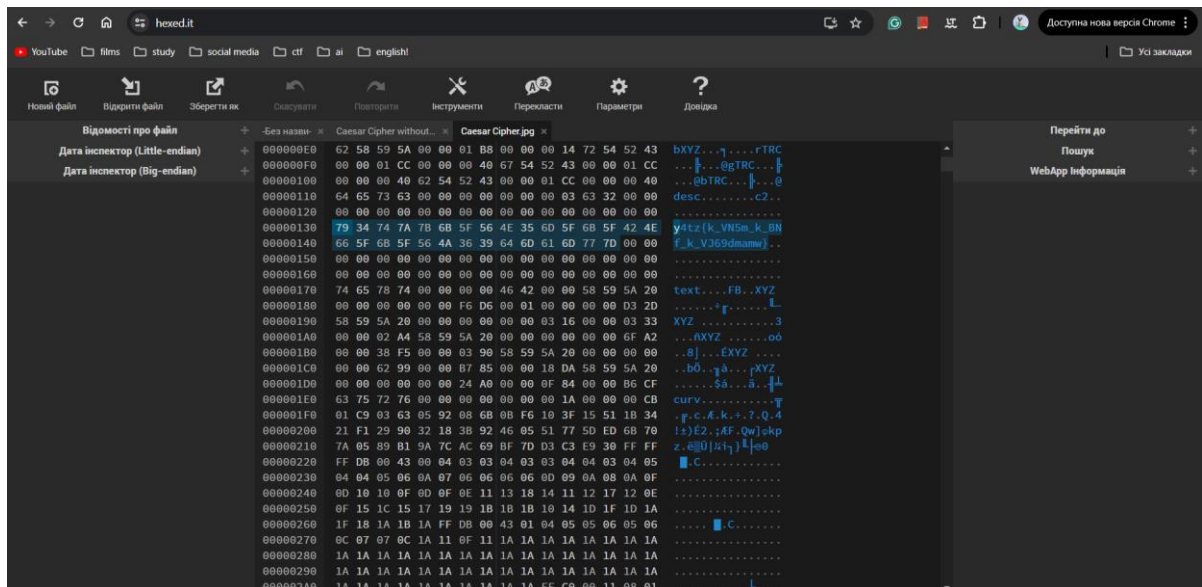


Рисунок 3.1 – Результат приховування прапора

Інструменти для рішення завдання: консольна команда strings, онлайн-інструмент aperisolve.fr для отримання прихованого прапора; сайт dcode чи cryptedii для розшифрування.

Тестування та перевірка:

1) Використаємо команду strings, яка виведе вміст файлу Caesar Cipher.jpg.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				75
Змн.	Арк.	№ докум.	Підпис	Дата		

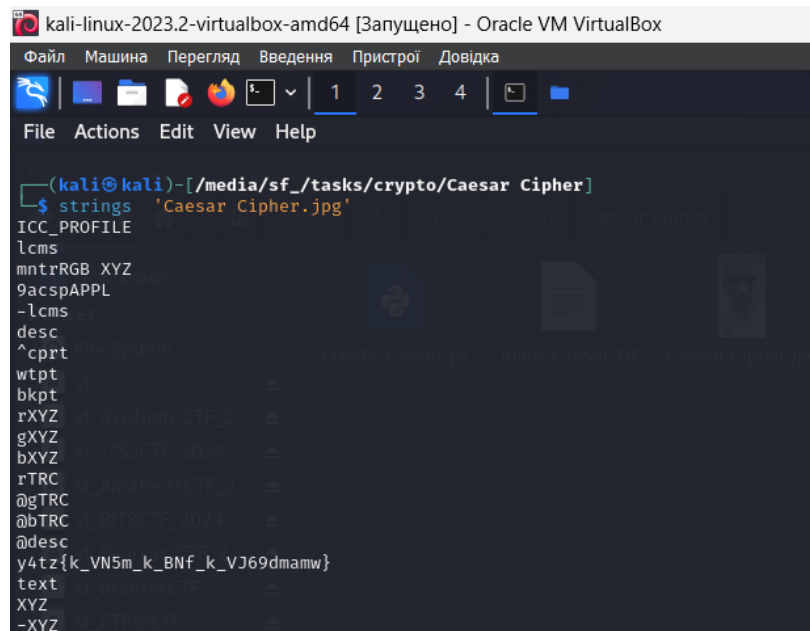


Рисунок 3.2 – Перегляд вмісту файлу

- 2) Бачимо рядок, який підозріло схожий за форматом на прапор:
y4tz{k_VN5m_k_BNf_k_VJ69dmamw}

Для його розшифрування можна написати відповідний код, або скористатися онлайн ресурсом. Як можна помітити наш прапор містить не лише літери, отже алфавіт для розшифрування буде складатися ще й з чисел.

Використаний алфавіт: abcdefghijklmnopqrstuvwxyz0123456789

Наступний крок – це підбір ключа. У нас є всього 35 можливих значень. Перебираючи по черзі всі варіанти, ми отримали ключ – 19.

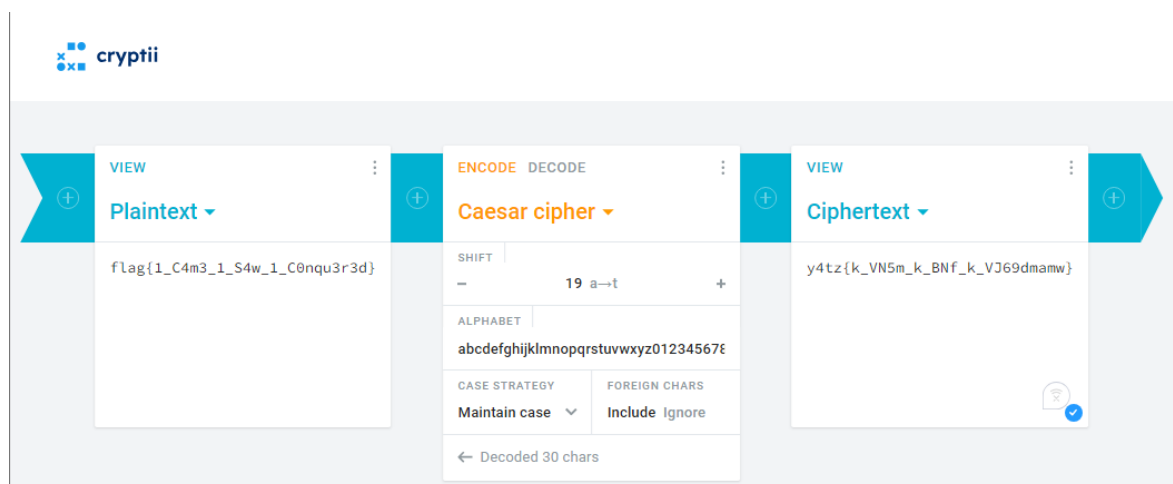


Рисунок 3.3 – Результат дешифрування прапора

Вихідні дані: flag{1_C4m3_1_S4w_1_C0nqu3r3d}

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		76

3.1.2 «Lovely Hill Picture»

Мета реалізації:

Це завдання зосереджене на ознайомленні учасників з ще одним класичним шифром – шифром Гілла. Варто уважно переглянути зображення та за допомогою отриманих даних розшифрувати прапор.

Використані техніки: шифр Гілла, стеганографія зображення

Мови програмування/інструменти: мова програмування Python, графічний редактор GIMP

Вхідні дані:

– ***Назва завдання:*** «Lovely Hill Picture»

– ***Опис до завдання:***

Одним із найвідоміших творів Томаса Гілла є картина «Індіанці біля озера у величному пейзажі Каліфорнії». Ця робота відображає не лише майстерність художника у вмінні передати красу природи та атмосферу, а й вбудовану в неї глибоку символіку.

Поглянувши на картину, спробуйте розшифрувати подане повідомлення.

– ***Формат прапора:*** після виконання завдання ви отримаєте рядок, який потрібно трохи змінити, щоб він відповідав формату `flag{СЛОВО1_СЛОВО2}`

– ***Файли для розв'язання завдання:*** `hill_landscape.jpg` та `hill_text.txt`

Кроки реалізації:

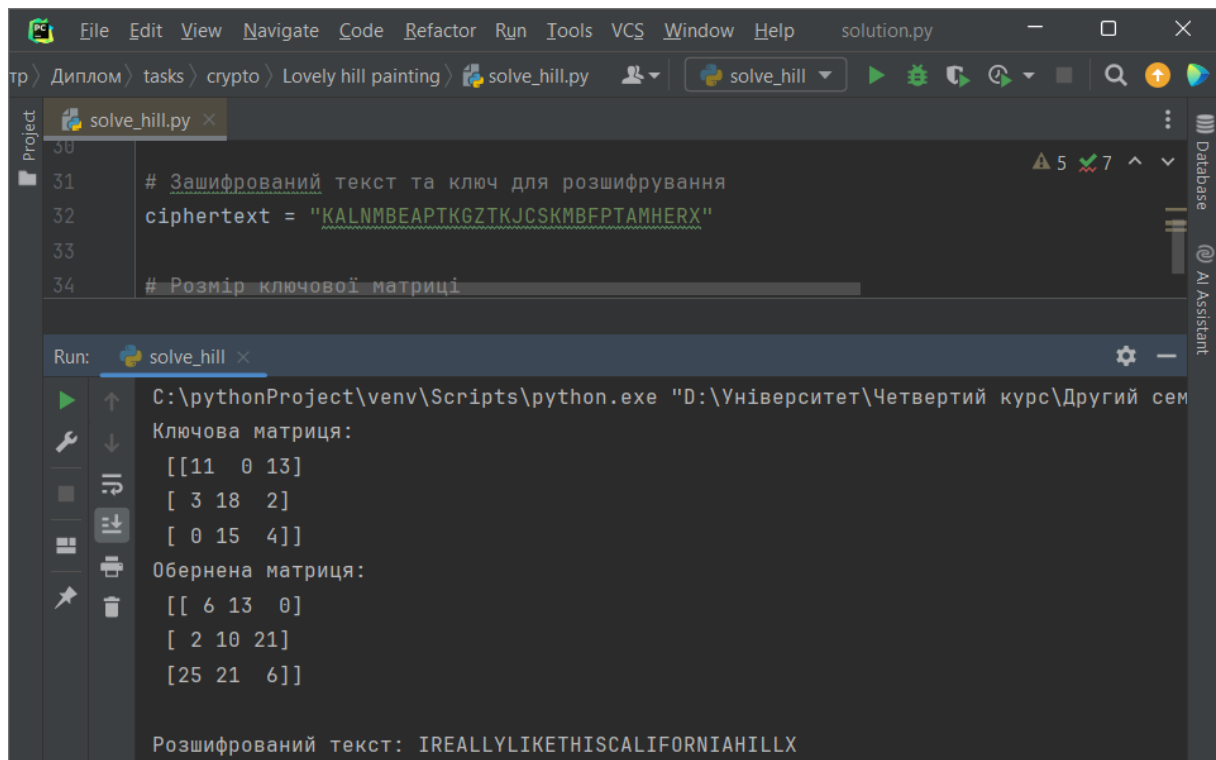
- 1) Обираємо рядок, який стане прапорцем.
- 2) Далі потрібно придумати ключ для шифрування довжиною 9 символів. Наприклад, слово LANDSCAPE.
- 3) Пишемо код для шифрування прапора шифром Гілла.
- 4) Зберігаємо зашифрований прапор у файлі `hill_text.txt`.
- 5) Використовуючи графічний редактор, додаємо напис з ключовою матрицею на зображення. Зберігаємо отриману картинку.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		77

Інструменти для рішення завдання: сайт dcode чи cсурTool для розшифрування повідомлення.

Тестування та перевірка:

1) Напишемо скрипт solve_hill.py для розшифрування прапорця.



```
31 # Зашифрований текст та ключ для розшифрування
32 ciphertext = "KALNMBEAPTKGZTKJCSKMBFPTAMHERX"
33
34 # Розмір ключової матриці
```

Run: solve_hill

C:\pythonProject\venv\Scripts\python.exe "D:\Університет\Четвертий курс\Другий сем

Ключова матриця:

```
[[11  0 13]
 [ 3 18  2]
 [ 0 15  4]]
```

Обернена матриця:

```
[[ 6 13  0]
 [ 2 10 21]
 [25 21  6]]
```

Розшифрований текст: IREALLYLIKETHISCALIFORNIAHILLX

Рисунок 3.4 – Результат дешифрування прапора

2) Розділимо слова отриманого тексту, щоб вони відповідали формату прапорця та видалимо символ X в кінці.

Вихідні дані: flag{I REALLY LIKE THIS CALIFORNIA HILL}

3.1.3 «Vigenère Quest»

Мета реалізації:

Завдання зосереджене на тому, щоб навчити учасників використовувати шифр Віженера для розшифрування тексту за допомогою ключового слова. Це завдання дозволяє застосувати знання про шифрування та дешифрування, а також розвиває логічне мислення і навички розв'язання криптографічних задач.

Використані техніки: шифр Віженера, стеганографія зображення

Мови програмування/інструменти: мова програмування Python, dcode.fr, Copilot

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		78

Вхідні дані:

- **Назва завдання:** «Vigenère Quest»
- **Опис до завдання:**

У далекому куточку музею, серед численних артефактів та експонатів, зберігається старовинна карта, яку приписують відомому мандрівникові й досліднику Блезу де Віженеру. Легенда свідчить, що ця карта містить секретне послання, яке відкриває шлях до прихованого скарбу. Щоб знайти коштовності, потрібно розшифрувати код, захований у карті.

Дослідники багато років намагалися розгадати цю загадку, але справжню суть тексту досі ніхто не розкрив. Відкрийте цей таємничий текст та дізнайтеся секрети, приховані у рядках старовинної карти.

- **Формат прапора:** після виконання завдання ви отримаєте рядок, який потрібно трохи змінити, щоб він відповідав формату `flag{слово1_слово2}`
- **Файли для розв'язання завдання:** `old-map.jpg`

Кроки реалізації:

- 1) Обираємо прапор для завдання.
- 2) Пишемо код, який зашифруємо список із 18 слів та останнього рядка, що є прапором. Ідея полягає в тому, що перше слово шифрується за допомогою ключа, а для кожного наступного слова ключем є попереднє незашифроване. Результат шифрування запишемо у файл `cipher.txt`.
- 3) Використаємо зображення старої мапи згенерованої штучним інтелектом як контейнер для текстового файлу.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		79

```

23     "code", "mystery", "journey", "artifacts", "adventure", "manuscri
24     "chronicle", "coordinates", "treasure",
25     "the treasure that you seek is hidden in the place you least expect"]
26
27 # Початковий ключ
28 key = "vigenere"

```

```

Run: create_vigenere
C:\pythonProject\venv\Scripts\python.exe "D:\Університет\Четвертий курс\Другий семестр\
Ciphertext 1 -> zfvbvvv
Ciphertext 2 -> lfsose
Ciphertext 3 -> ywxwi
Ciphertext 4 -> vljxzhchr
Ciphertext 5 -> wbrvhb
Ciphertext 6 -> vmutsowva
Ciphertext 7 -> omygby
Ciphertext 8 -> lrimrqe
Ciphertext 9 -> cbfm
Ciphertext 10 -> omvxgfb
Ciphertext 11 -> vmmkrvw
Ciphertext 12 -> jfnzseacg
Ciphertext 13 -> auomstkwk
Ciphertext 14 -> mdiyfvlttv
Ciphertext 15 -> oozjsuj
Ciphertext 16 -> evddnauns
Ciphertext 17 -> evffqqlxgz
Ciphertext 18 -> vfsrvcee
Ciphertext 19 -> myi tjyrwnii tzuk chl wewe zw azhdwh zr myi pduti rfy lwujx xoteun
Encrypted words saved to cipher.txt

```

Рисунок 3.5 – Результат виконання коду

```

C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.22631.3593]
(с) Корпорація Майкрософт. Усі права захищені.

D:\Університет\Четвертий курс\Другий семестр\Диплом\tasks\crypto\Vigenère Qu
est>copy /b old-map(original).jpg + mystery.zip old-map.jpg
old-map(original).jpg
mystery.zip
        1 file(s) copied.

D:\Університет\Четвертий курс\Другий семестр\Диплом\tasks\crypto\Vigenère Qu
est>

```

Рисунок 3.6 – Результат об'єднання файлів

Інструменти для рішення завдання: binwalk, aperisolve, мова програмування Python, dcode.fr

Тестування та перевірка:

- 1) Переглянемо вміст зображення мапи за допомогою онлайн-платформи aperisolve.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				80
Змн.	Арк.	№ докум.	Підпис	Дата		

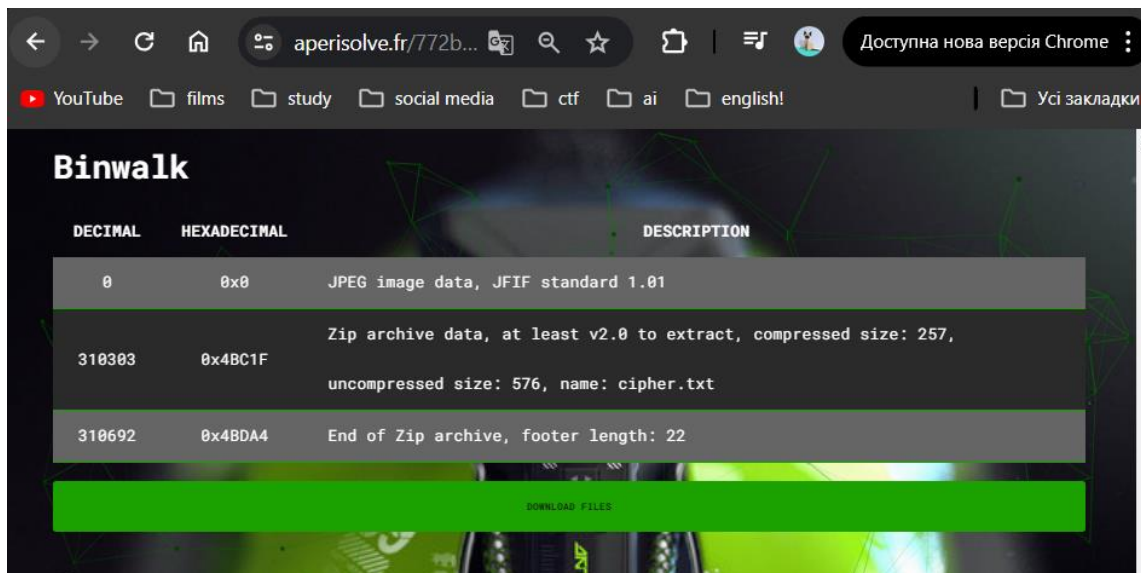


Рисунок 3.7 – Перегляд вмісту зображення

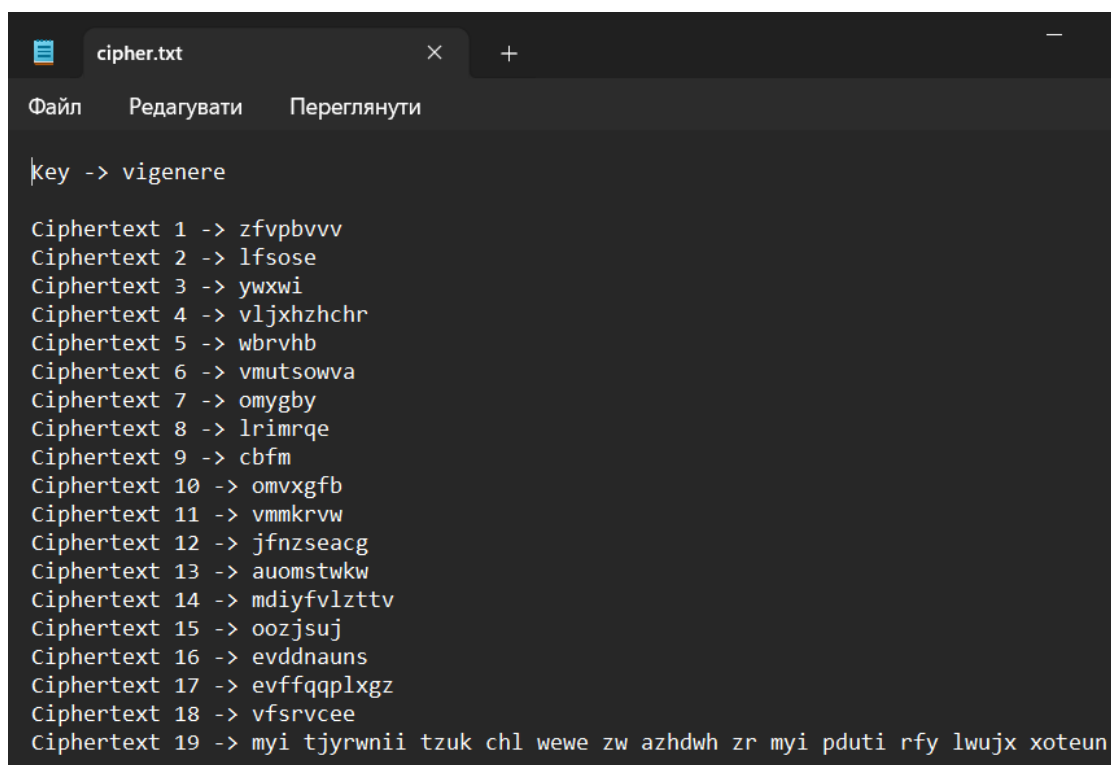


Рисунок 3.8 – Вміст текстового файлу

- Для розшифрування слів напишемо код або використаємо онлайн-інструменти dcode.fr.

```

44 with open("cipher.txt", "r") as file:
45     lines = file.readlines()
46
47 # Читання ключа

Run: solve_vigenere
C:\pythonProject\venv\Scripts\python.exe "D:\Університет\Четвертий курс\Другий семестр\
Plaintext 1 -> explorer
Plaintext 2 -> hidden
Plaintext 3 -> route
Plaintext 4 -> expedition
Plaintext 5 -> secret
Plaintext 6 -> discovery
Plaintext 7 -> legend
Plaintext 8 -> ancient
Plaintext 9 -> code
Plaintext 10 -> mystery
Plaintext 11 -> journey
Plaintext 12 -> artifacts
Plaintext 13 -> adventure
Plaintext 14 -> manuscripts
Plaintext 15 -> compass
Plaintext 16 -> chronicle
Plaintext 17 -> coordinates
Plaintext 18 -> treasure
Plaintext 19 -> the treasure that you seek is hidden in the place you least expect

Flag: flag{the_treasure_that_you_seek_is_hidden_in_the_place_you_least_expect}

```

Рисунок 3.9 – Результат розшифрування слів

Вихідні дані:

flag{the_treasure_that_you_seek_is_hidden_in_the_place_you_least_expect}

3.2 Практична реалізація криптографічних завдань середнього та складного рівнів

3.2.1 «D1v3 Into Th3 D3pths!»

Мета реалізації:

Завдання зосереджене на використанні різних простих шифрів та систем кодування. Учасникам необхідно застосувати свої знання про шифрування та кодування, а також проявити уважність, щоб розкрити таємниці.

Використані техніки: ROT13, ASCII, Binary, Base64, Нех, Атбаш шифр, Base62.

Мови програмування/інструменти: мова програмування Python, CyberChef, сайти <https://www.rapidtables.com/>, <https://www.duplchecker.com/binary-to-ascii.php>, <https://www.base64encode.org/>

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				82
Змн.	Арк.	№ докум.	Підпис	Дата		

Вхідні дані:

- **Назва завдання:** «D1v3 Into Th3 D3pths!»
- **Опис до завдання:**

У музеї відкрилася нова експозиція, присвячена морським скарбам, знайденим на затонулих кораблях. Серед експонатів є дивовижна мушля, яка, за легендою, приховує у собі важливий секрет.

Ваша задача - дослідити цю мушлю та знайти приховане повідомлення, використовуючи свої навички криптографії. Пам'ятайте, що справжні скарби моря не завжди є тими, які здаються на перший погляд!

- **Складність:** середній рівень
- **Формат прапора:** після виконання завдання ви отримаєте прапор формату flag{Слово1_Слово2}
- **Файли для розв'язання завдання:** shell.txt
- **Підказки:**

Підказка №1. Як перлина в мушлі, так і прапор знайдеш в її закрученій формі.

Кроки реалізації:

- 1) Перш за все, обираємо рядок для прапора, який розміщуємо у формі закрученої мушлі.
- 2) Потім застосовуємо простий підстановочний шифр ROT13, який замінює кожен літер на ту, що стоїть через 13 позицій після неї в алфавіті.
- 3) Наступний крок – це використання кодування ASCII, після цього Binary, Base64, Hex, шифр Атбаш і Base62. На різних етапах додаємо ще підказки.
- 4) Збережемо отриманий результат в окремий текстовий файл.

Інструменти для рішення завдання: мова програмування Python, CyberChef сайти <https://www.rapidtables.com/>, <https://www.duplichecker.com/binary-to-ascii.php>, <https://www.base64decode.org/>

Тестування та перевірка:

- 1) Для перевірки послідовно дешифруємо чи декодуємо вміст файлу.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		83

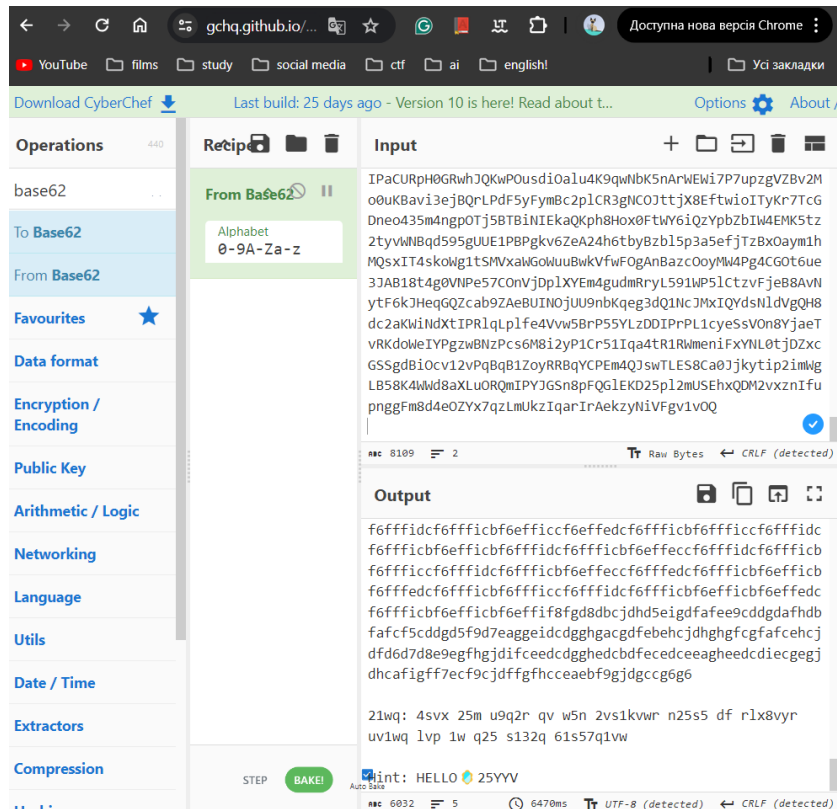


Рисунок 3.10 – Результат декодування Base62

- 2) Підказка натякає нам на, те було використано дзеркальний шифр або ще відомий як шифр Атбаш. Для дешифрування оберемо алфавіт з додаванням цифр: ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789.

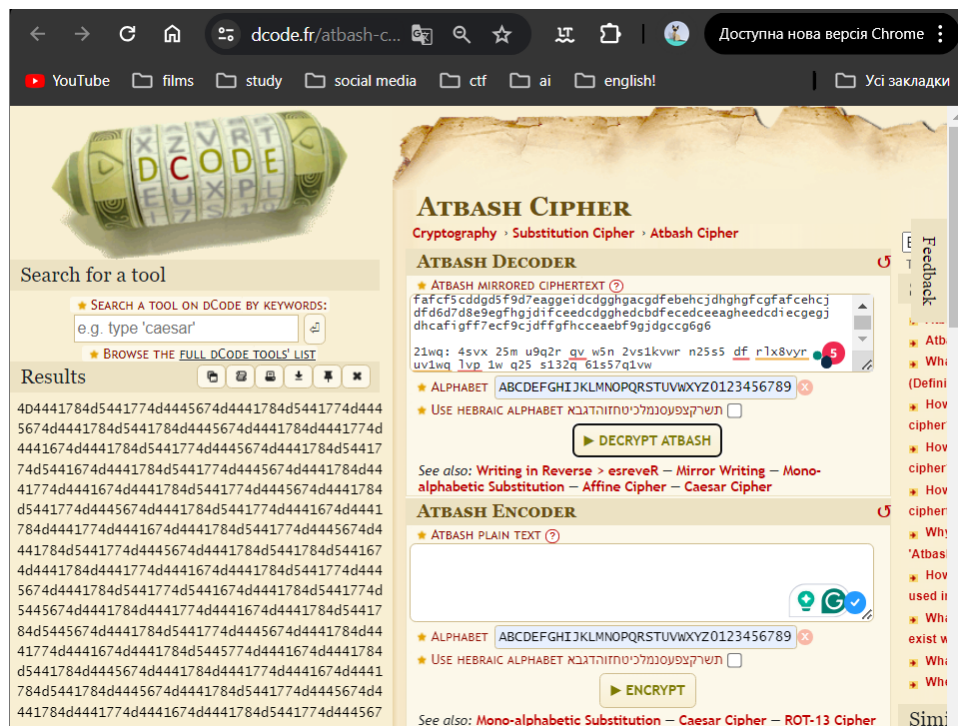


Рисунок 3.11 – Результат дешифрування

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				84
Змн.	Арк.	№ докум.	Підпис	Дата		

- 3) Щоб визначити, яке кодування чи шифрування було застосовано до отриманого повідомлення, скористаємося Cipher Identifier. Цей онлайн-інструмент надасть нам перелік можливих шифрів.



Рисунок 3.12 – Результат аналізу тексту

- 4) Послідовно декодуємо наш шифротекст. Спершу Hex, потім Base64, Binary і ASCII.

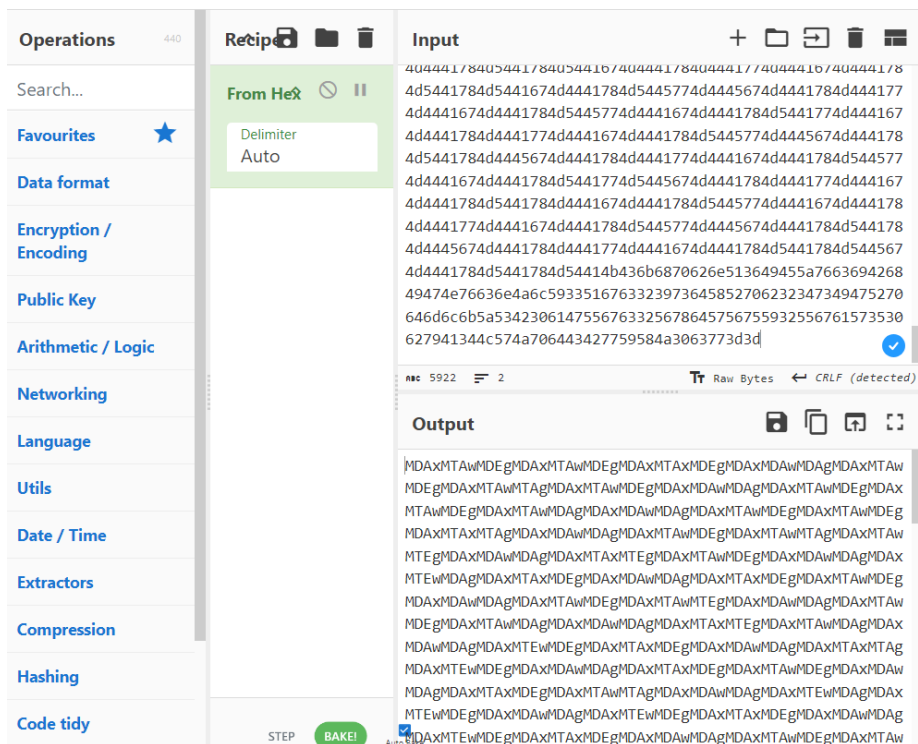


Рисунок 3.13 – Результат декодування Hex

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				85
Змн.	Арк.	№ докум.	Підпис	Дата		

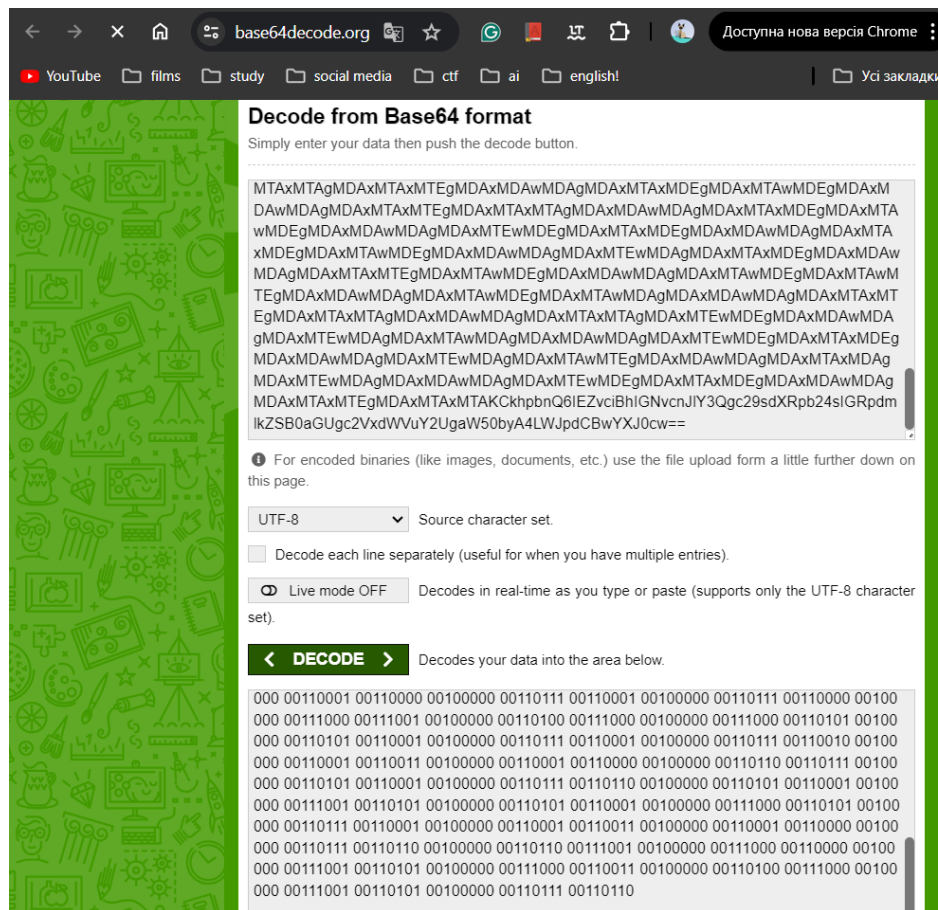


Рисунок 3.14 – Результат декодування Base64

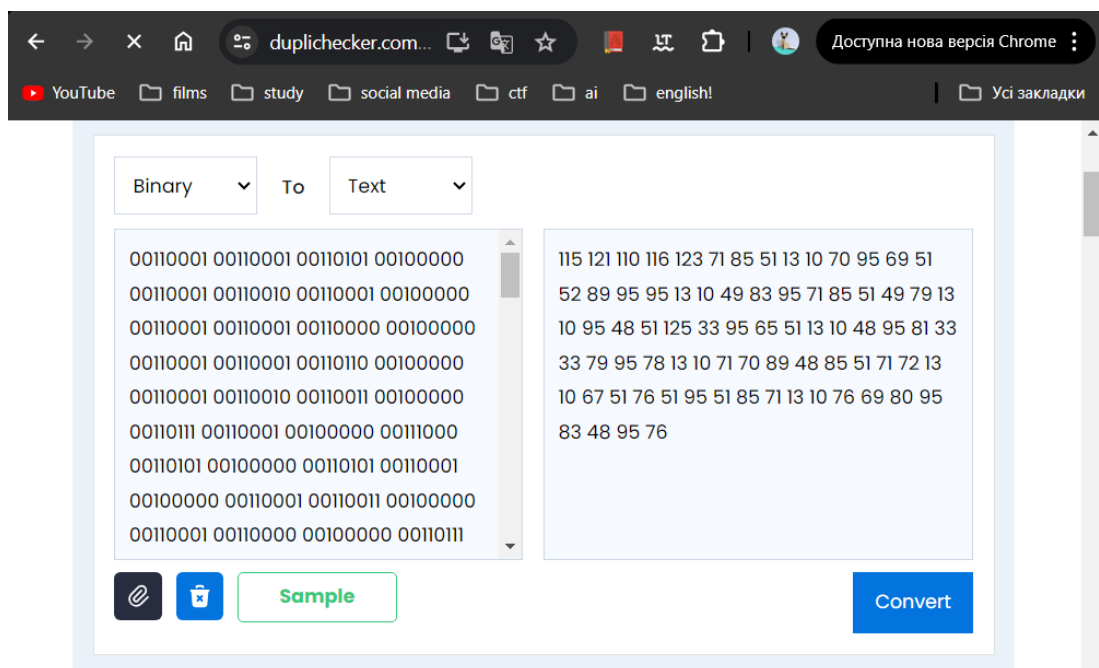


Рисунок 3.15 – Результат декодування Binary

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				86
Змн.	Арк.	№ докум.	Підпис	Дата		

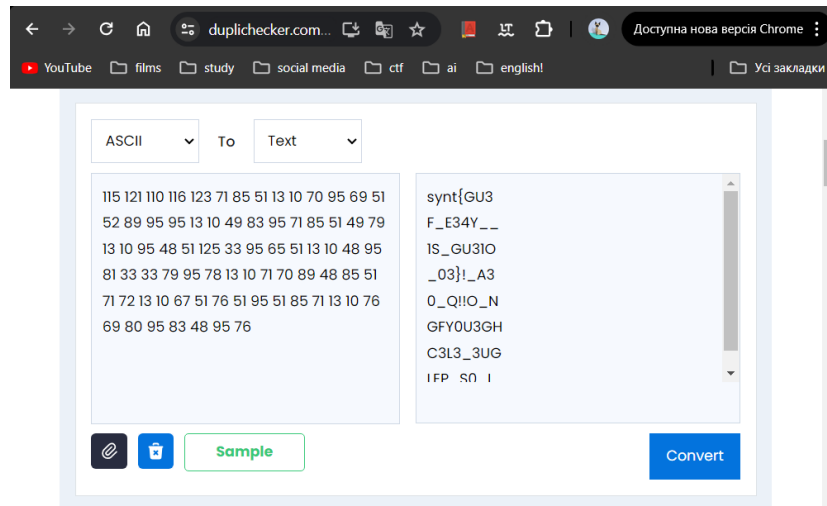


Рисунок 3.16 – Результат декодування ASCII

- 5) Отриманий текст, схоже, зашифрований простим шифром. Знову скористаємося Cipher Identifier, щоб визначити, яким саме.



Рисунок 3.17 – Результат аналізу шифротексту

- 6) Останній етап – це дешифрування прапора за допомогою ROT13.

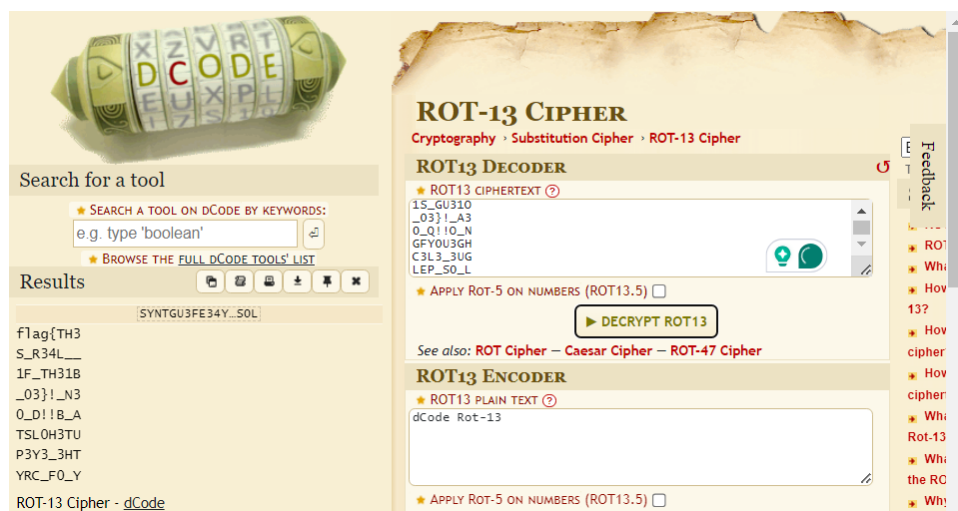


Рисунок 3.18 – Результат дешифрування ROT13

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				87
Змн.	Арк.	№ докум.	Підпис	Дата		

Вихідні дані:

flag{TH3_B3AUTY_0F_CRYPT0_1S_R34L_1N_TH3_3Y3S_0F_TH3_B3H0L
D3R}

3.2.2 «The Starry Bytes»

Мета реалізації:

Перш за все, це завдання передбачає, що учасники ознайомлені із магічними байтами формату PNG та розуміють як працює операція XOR. Використавши вхідні дані та свої знання для написання коду, вони отримають прапор.

Використані техніки: операція XOR

Мови програмування/інструменти: графічний редактор GIMP, мова програмування Python

Вхідні дані:

- ***Назва завдання:*** «The Starry Bytes»
- ***Опис до завдання:***

Картина, здається була пошкоджена якимсь дивним чином. Однак, поглянувши уважніше ми можна помітити такий напис: «Спостереження за байтами зображення із зірками на небі схоже на глибоке занурення у космічну глибину. Так само, як кожна зірка має свою історію та значення, байт зображення несе в собі інформацію, яка розкривається лише тим, хто вміє читати цей космічний код».

Вам потрібно дослідити файли та знайти спосіб відновити картину.

- ***Складність:*** середній рівень
- ***Формат прапора:*** після виконання завдання ви отримаєте прапор формату flag{Слово1_Слово2}
- ***Файли для розв'язання завдання:*** source.txt, encrypted_stars.txt

Кроки реалізації:

- 1) Вигадуємо прапором, що за тематикою підходить до завдання.
- 2) За допомогою графічного редактора додаємо до картини напис з прапором.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		88

3) Далі пишемо скрипт create_stars.py, що шифрує зображення, виконуючи операцію XOR з ключем. Як відомо перші вісім байтів файлу PNG завжди містять такі значення: 89 50 4e 47 0d 0a 1a 0a. Їх ми і обираємо у ролі ключа. В результаті отримуємо файл encrypted_stars.txt.

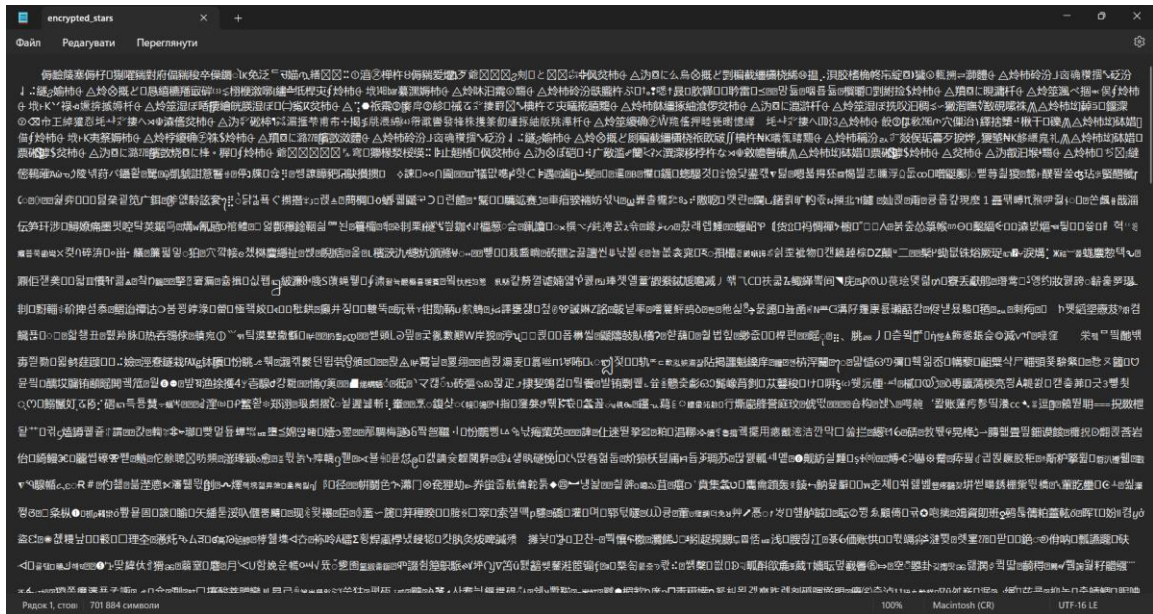


Рисунок 3.19 – Вміст файлу encrypted_stars.txt

Інструменти для рішення завдання: мова програмування Python

Тестування та перевірка:

Для вирішення завдання напишемо скрипт solve_stars.py, який виконає операцію XOR між першими 8 байтами зображення формату PNG та вмістом файлу encrypted_stars.txt.

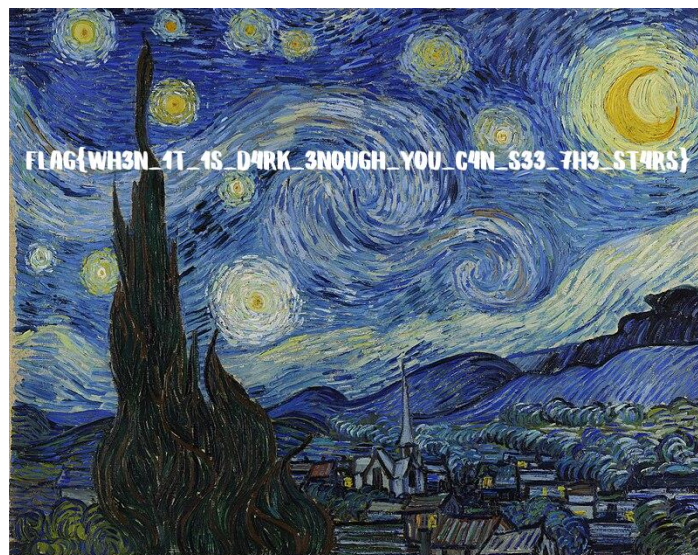


Рисунок 3.20 – Результат дешифрування зображення

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				89
Змн.	Арк.	№ докум.	Підпис	Дата		

Вихідні дані: flag{Wh3n_1t_1s_D4rk_3n0ugh_Y0u_C4n_S33_7h3_St4rs}

3.2.3 «Rhythmic Symphony of Art»

Мета реалізації:

Перш за все, завдання зосереджене на ознайомленні учасників з криптографічним алгоритмом RSA. Для того, щоб трохи ускладнити роботу, вхідні дані розміщено на сайті для онлайн зберігання тексту. Посилання на цей веб-ресурс закодовано за допомогою коду Морзе, що додає додатковий рівень складності та вимагає від учасників вміння працювати з різними методами шифрування чи кодування. Завдання включає не лише декодування азбуки Морзе, але й роботу з самим алгоритмом RSA для розшифрування отриманих даних.

Використані техніки: криптографічний алгоритм RSA, азбука Морзе

Мови програмування/інструменти: мова програмування Python, інструменти для створення коду Морзе, наприклад, cryptii.com; сайт <https://badcord.ct8.pl/>

Вхідні дані:

- ***Назва завдання:*** «Rhythmic Symphony of Art»
- ***Опис до завдання:***

Одним із найцікавіших експонатів цього музею є унікальний нотний лист. Ви можете запитати, чим саме він незвичний? Його особливість полягає в тому, що замість звичних для всіх нот, музика на цьому аркуші написана за допомогою крапок та тире. Скільки б людей не намагалися «зіграти» цю мелодію, жодному музикантові не вдалося відтворити звучання так, як воно задумувалося. Ходять легенди, що тому, хто відтворить мелодію, відкриється справжній скарб. Деякі вважають, що ця мелодія є ключем до розкриття великої таємниці або навіть відкриття невідомих досі знань.

Спробуйте і ви свої сили в розгадуванні цієї музичної таємниці. Хто знає, можливо, саме вам вдасться знайти ключ до її розуміння. Уявіть собі відчуття триумфу, коли ви розкриєте мелодію, яка залишалася нерозгаданою протягом століть, і знайдете шлях до скарбу, який був мрією багатьох поколінь.

- ***Складність:*** середній рівень

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		90

- **Формат прапора:** після виконання завдання ви отримаєте прапор формату `flag{Слово1_Слово2}`
- **Файли для розв'язання завдання:** `symphony.txt`
- **Підказки:**

Підказка №1. Всі шукають відповідь серед звичних символів, але справжня музика криється в тонкостях нижнього регістру. Можливо, саме там ви знайдете ключ до скарбу.

Кроки реалізації:

- 1) Обираємо креативний рядок для прапорця.
- 2) Пишемо код для шифрування прапора за допомогою криптографічного алгоритму RSA.
- 3) Важливим ще кроком є додавання простих дільників (p та q) числа n на сайт <http://factordb.com/>. Це зроблено для того, щоб полегшити учасникам процес розшифрування прапора.

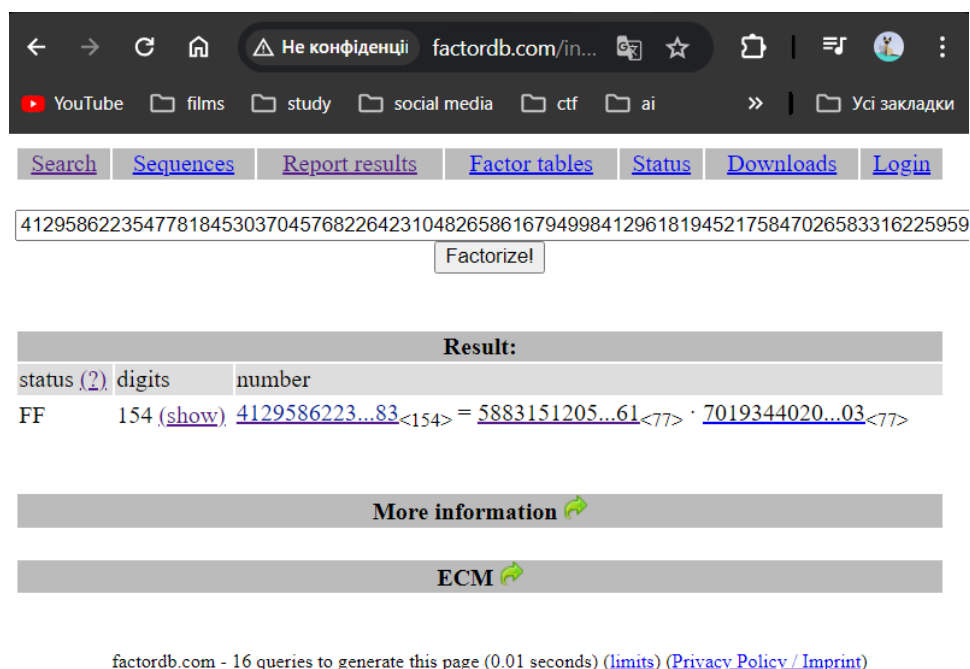


Рисунок 3.21 – Результат додавання простих дільників

- 4) Далі розмістимо значення n , e та $cipher$ на веб-сайті для онлайн зберігання тексту протягом певного періоду часу. Наприклад, `pastebin`, `rlim` чи `badcord`.

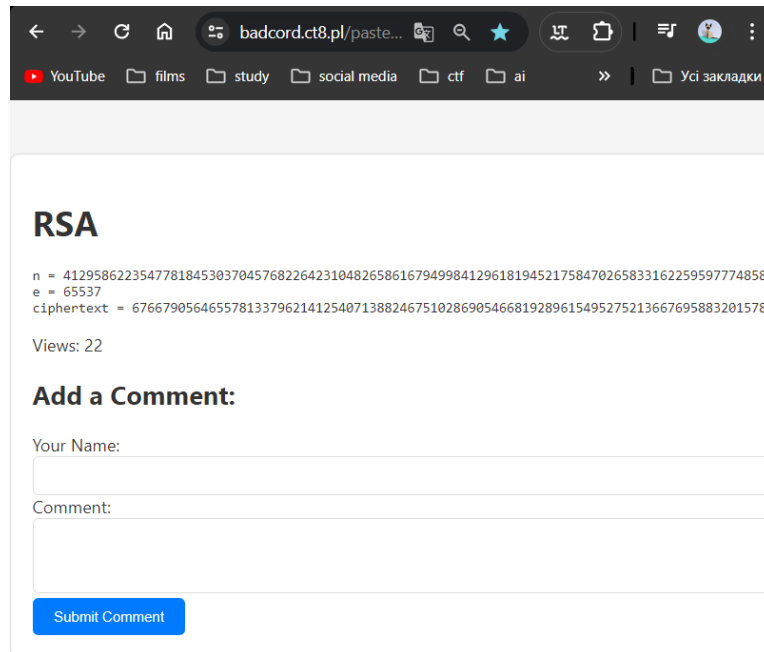


Рисунок 3.22 – Результат розміщення даних RSA на сайті

5) Скопіюємо посилання та закодуємо його за допомогою коду Морзе.

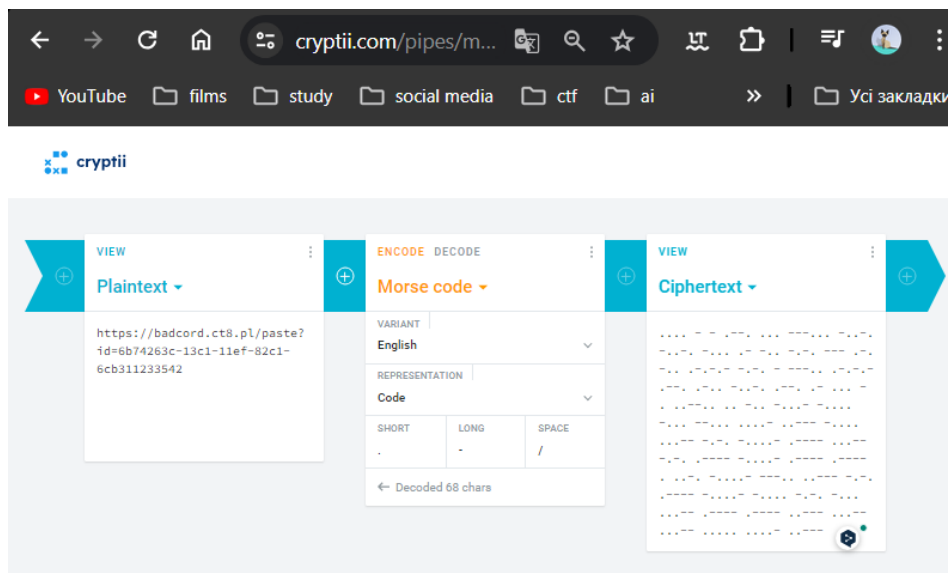


Рисунок 3.23 – Результат кодування даних азбукою Морзе

Інструменти для рішення завдання: інструменти для декодування Морзе коду, наприклад, cryptii.com; мова програмування Python, factordb чи утиліти RsaCtfTool для дешифрування прапора.

Тестування та перевірка:

Перевіримо дешифрування прапора за допомогою утиліти RsaCtfTool. Як бачимо процес досить простий завдяки тому, що factordb містить інформацію про прості дільники числа n .

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				92
Змн.	Арк.	№ докум.	Підпис	Дата		

3.2.4 «Modern Art»

Мета реалізації:

Познайомити учасників з таким криптографічним методом, як візуальна криптографія. Для цього варто дослідити декілька частин зображення, щоб знайти спосіб поєднати їх у єдину картинку.

Використані техніки: візуальна криптографія

Мови програмування/інструменти: Paint, мова програмування Python

Вхідні дані:

- *Назва завдання:* «Modern Art»
- *Опис до завдання:*

Сучасне мистецтво сповнене суперечностей. Те, що вважається красивим у традиційному мистецтві, може кардинально відрізнятися від сучасних естетичних ідеалів. Модерне мистецтво може бути провокаційним, дискомфортним або навіть навмисно некрасивим, щоб передати певне повідомлення або викликати емоційну реакцію.

Вам варто ознайомитися з картинами, щоб зрозуміти, яке саме послання прагнув передати автор через свої роботи.

- *Складність:* складний рівень
- *Формат прапора:* після виконання завдання ви отримаєте прапор формату flag{Слово1_Слово2}
- *Файли для розв'язання завдання:* painting1.png, painting2.png
- *Підказки:*

Підказка №1. Кажуть джерелом натхнення для цих картин стала праця Мона Наора та Аді Шаміра.

Кроки реалізації:

- 1) Обираємо рядок, який стане прапором та в Paint створюємо картинку з ним.
- 2) Пишемо код на мові програмування Python для того, зашифрувати зображення і отримати дві його частини.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		94

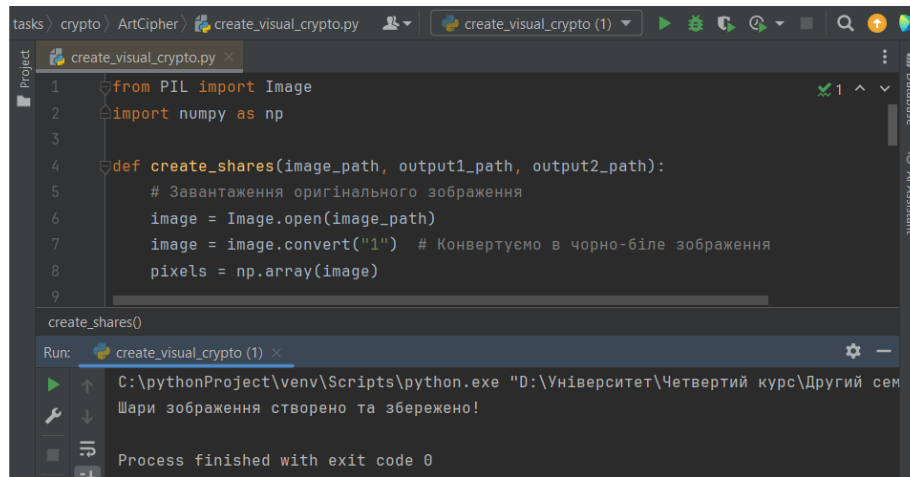


Рисунок 3.26 – Результат виконання коду

Інструменти для рішення завдання: мова програмування Python

Тестування та перевірка:

- 1) Для тестування напишемо зворотній код, який поєднає шари зображення.

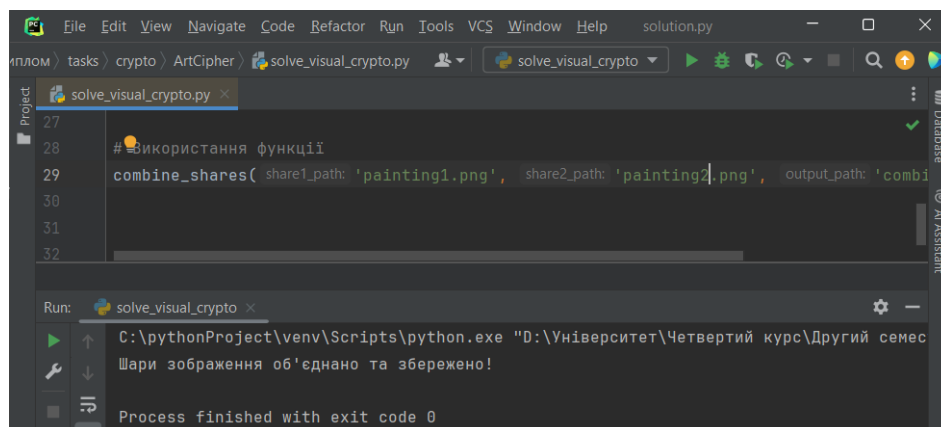


Рисунок 3.27 – Результат виконання коду

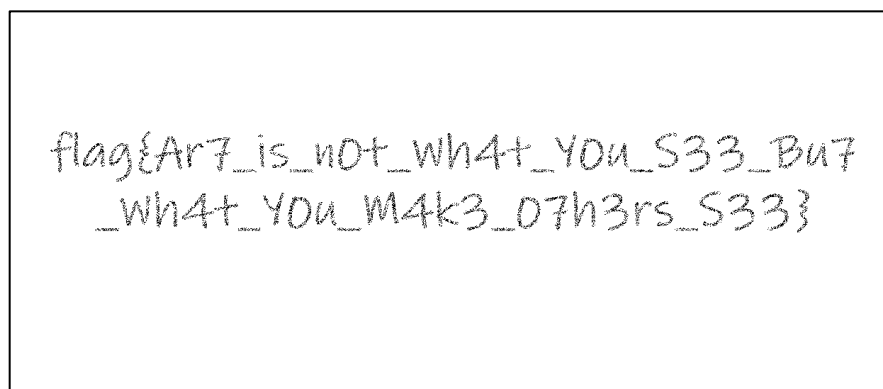


Рисунок 3.28 – Отримане зображення з прапором

Вихідні дані:

flag{Ar7_is_n0t_Wh4t_Y0u_S33_Bu7_Wh4t_Y0u_M4k3_07h3rs_S33}

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		95

3.2.5 «Art Enthusiasts' Seminar»

Мета реалізації:

Завдання зосереджене на алгоритмі шифрування AES. Учасникам варто проаналізувати подані файли та використавши свої навички з криптографії та програмування на Python, розшифрувати прапор.

Використані техніки: алгоритм шифрування AES

Мови програмування/інструменти: мову програмування Python

Вхідні дані:

- *Назва завдання:*
- *Опис до завдання:*

Час від часу в музеї відбувається семінар для поціновувачів мистецтва. Люди відвідують його, щоб дізнатися більше про різні художні стилі, техніки та історію мистецтва. На таких заходах зазвичай присутні експерти в галузі, які проводять лекції та майстер-класи, відповідають на запитання учасників, а також організовують обговорення мистецьких творів.

На одному з семінарів обговорення творів мистецтва пролунала цікава думка про те, що створення картин подібне до шифрування, адже митець вкладає свої ідеї, почуття та думки у різноманітні символи, кольори та форми. У цьому справді є щось особливе.

Ваша задача – переглянути подані файли та розшифрувати прапор.

- *Складність:* складний рівень
- *Формат прапора:* після виконання завдання ви отримаєте прапор формату flag{Слово1_Слово2}
- *Файли для розв'язання завдання:* encrypt.py, output..txt

Кроки реалізації:

- 1) Підбираємо рядок, який підійде на роль прапора.
- 2) Пишемо код для шифрування прапорця алгоритмом шифрування AES. Зашифроване повідомлення разом із сіллю та вектором ініціалізації зберігається до текстового файлу.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		96

```

11 password = 'artist10'.encode('utf-8') # Пароль
12 salt = get_random_bytes(16) # Генерація випадкової солі
13 iv = get_random_bytes(16) # Генерація випадкового IV
14 message = 'flag{AES_3ncRyp710n_15_4N_4r7_f0r_M45T3r_C0d3r5}'.encode('utf-8') # По
15
16 # Генерація ключа за допомогою PBKDF2

```

```

Run: create_aes x
C:\pythonProject\venv\Scripts\python.exe "D:\Університет\Четвертий курс\Другий сем
Зашифроване повідомлення (base64): M34r1PCg8zBXnX3MjfPcw18dp41kWuRDkYbZ9osk97RtRqW
Сіль (base64): 8PFa1MV3U6fqyeQUOW1S7g==
IV (base64): dG7se1lvvU909cvxual1Ww==
Process finished with exit code 0

```

Рисунок 3.29 - Результат виконання коду

Інструменти для рішення завдання: мову програмування Python

Тестування та перевірка:

Маємо два файли: один – це скрипт для шифрування, інший – зашифровані дані. Здається, цього достатньо для розшифрування прапора, проте не вистачає паролю, з якого генерується ключ. Для цього потрібно застосувати метод підбору паролю. Напишемо код для дешифрування.

```

18 password_file = 'rockyou.txt' # Файл зі списком можливих паролів
19
20 # Відкриття файлу з паролями та перебір можливих паролів
21 with open(password_file, 'r') as f:
22     for line in f:
23         password = line.strip().encode('utf-8')

```

```

Run: solve_aes_brute_force_password x
C:\pythonProject\venv\Scripts\python.exe "D:\Університет\Четвертий курс\Другий семес
Знайдено пароль: artist10
Розшифроване повідомлення: flag{AES_3ncRyp710n_15_4N_4r7_f0r_M45T3r_C0d3r5}
Process finished with exit code 0

```

Рисунок 3.30 - Результат виконання коду

Виконання коду тривало приблизно 12 хвилин.

Вихідні дані: flag{AES_3ncRyp710n_15_4N_4r7_f0r_M45T3r_C0d3r5}

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		97

3.3 Практична реалізація стеганографічних завдань легкого рівня

3.3.1 «The Winged Victory of Samothrace»

Мета реалізації:

Ознайомити учасників з різними інструментами перегляду метаданих: від консольних команд до відповідних сайтів, а також використати конвертування ASCII значення в текст.

Використані техніки: стеганографія зображення, зміна метаданих, кодування ASCII

Мови програмування/інструменти: мова програмування Python

Вхідні дані:

- **Назва завдання:** «The Winged Victory of Samothrace»
- **Опис до завдання:**

Перед вами «Вікторія Самофракійська» або «Ніка Самофракійська» — це давньогрецька скульптура, створена приблизно між 220 і 190 роками до н.е. Вона зображує Ніку, богиню перемоги, яка стоїть на кораблі із спущеними крилами, готова пролетіти у небо. Ця скульптура є однією з найвідоміших реліквій давньогрецького мистецтва і представляє найвищі досягнення художньої майстерності Греції в епоху Еллінізму.

Огляньте уважно скульптуру, здається, вона містить в собі багато таємниць.

- **Формат прапора:** після виконання завдання ви отримаєте прапор формату flag{Слово1_Слово2}
- **Файли для розв'язання завдання:** зображення The_Winged_Victory_of_Samothrace.jpg

Кроки реалізації:

- 1) Обираємо рядок, що буде прапором для цього завдання.
- 2) Пишемо код, який спершу конвертує текст в ASCII, а далі додає коментар з прапором до метаданих картини.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		98

Інструменти для рішення завдання: ExifTool, aperiolve або інший онлайн-ресурс для перегляду метаданих, наприклад, IMGonline.com.ua.

Тестування та перевірка:

- 1) Використовуємо команду exiftool для перегляду метаданих зображення The_Winged_Victory_of_Samothrace.jpg

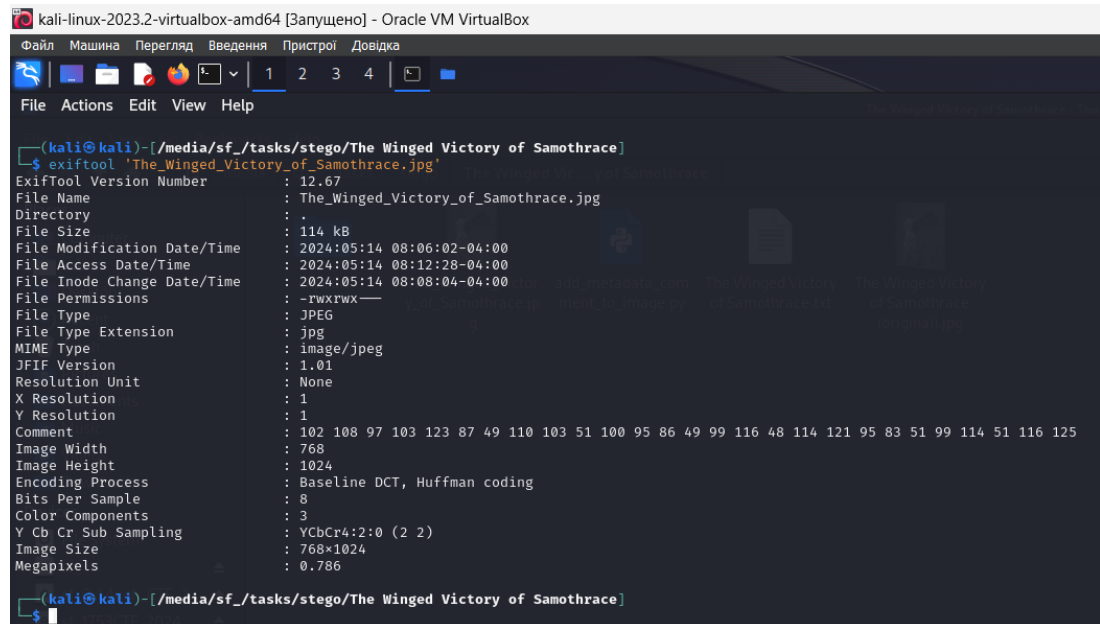


Рисунок 3.31 – Перегляд метаданих зображення

Бачимо коментар, доданий за допомогою коду.

Comment: 102 108 97 103 123 87 49 110 103 51 100 95 86 49 99 116 48 114 121 95 83 51 99 114 51 116 125

- 2) Використаємо онлайн-ресурс для переведення значень ASCII в текст.

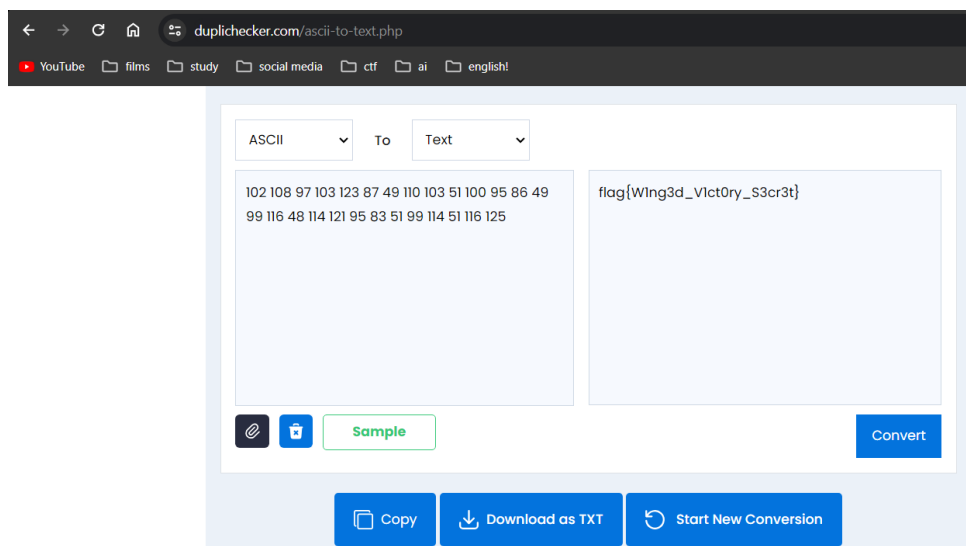


Рисунок 3.32 – Отриманий прапор

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				99
Змн.	Арк.	№ докум.	Підпис	Дата		

Вихідні дані: flag{W1ng3d_V1ct0ry_S3cr3t}

3.3.2 «Unveiling the Invisible Art»

Мета реалізації:

Ознайомити учасників із технікою текстової стеганографії Zero-Width Characters. Завдання передбачає використання різних інструментів для того, щоб отримати приховане повідомлення.

Використані техніки: zero-width characters (символи нульової ширини)

Мови програмування/інструменти: [онлайн ресурс](#) для приховання прапора

Вхідні дані:

– ***Назва завдання:*** «Unveiling the Invisible Art»

– ***Опис до завдання:***

Відвідувачі, мандруючи між експонатами, раптом помітили дивний лист, який хтось, мабуть, випадково залишив на полиці поруч із стародавніми сувіями.

Цей лист здавався дуже старим та зношеним, і на перший погляд нічого особливого в ньому не було. Але один з гостей, який був фанатом таємничих історій, вирішив розглянути його уважніше і помітив дещо цікаве.

Дослідіть зміст листа та розкрийте його таємницю.

– ***Формат прапора:*** після виконання завдання ви отримаєте прапор формату flag{Слово1_Слово2}

– ***Файли для розв'язання завдання:*** letter_to_my_friend.txt

Кроки реалізації:

- 1) Створюємо прапорець та вигадуємо зміст листа відповідно до теми.
- 2) Переходимо на відповідний сайт, щоб приховати прапор між рядками листа. Зберігаємо отриманий текст.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		100

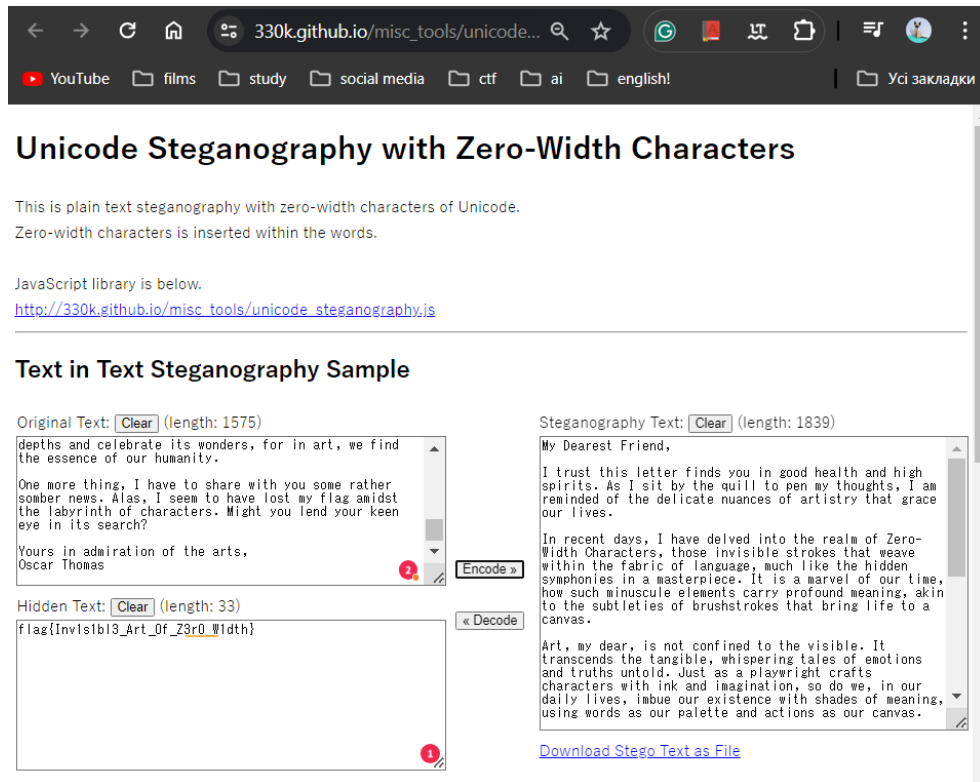


Рисунок 3.33 – Результат приховування прапора в тексті

Інструменти для рішення завдання: можна використати аналогічний [сайт](#), яким було приховано повідомлення.

Тестування та перевірка:

Перевірку ми проводили на тому ж сайті, що використовувався раніше.

Text in Text Steganography Sample

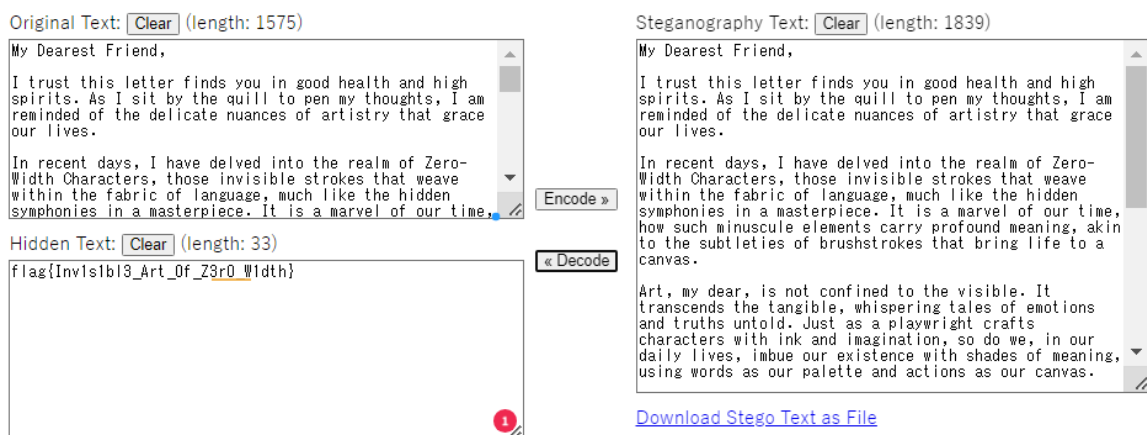


Рисунок 3.34 – Результат виявлення прапора в тексті

Вихідні дані: flag{Inv1s1bl3_Art_Of_Z3r0_W1dth}

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				101
Змн.	Арк.	№ докум.	Підпис	Дата		

3.3.3 «Black Square»

Мета реалізації:

Завдання передбачає, що учасники ознайомляться кольоровою моделлю RGB та за допомогою цих знань знайдуть прихований на картині напис.

Використані техніки: стеганографія зображення, RGB канали

Мови програмування/інструменти: графічний редактор GIMP

Вхідні дані:

– *Назва завдання:* «Black Square»

– *Опис до завдання:*

«Чорний квадрат» - картина українського художника-авангардиста Казимира Малевича, виконана в стилі супрематизму. Вона була створена в 1915 році і представляє собою абстрактний квадрат, який зображений на білому фоні. Хоча це виглядає дуже просто, «Чорний квадрат» має глибокий символічний зміст і відіграв важливу роль у мистецькому руху того часу.

У результаті рентгенофлуоресцентного та інфрачервоного аналізу, роботи з мікроскопом та інших сучасних досліджень було виявлено, що під шаром «Чорного квадрата» опинилися дві композиції Малевича.

Проведіть власне дослідження, щоб виявити що ще приховує ця картина під шарами пікселів.

– *Формат прапора:* після виконання завдання ви отримаєте прапор формату flag{Слово1_Слово2}

– *Файли для розв'язання завдання:* зображення black_square.jpg

– *Підказки:*

Підказка №1. Чи чули ви колись про окуляри з кольоровими лінзами? Надягніть їх і погляньте на картину ще раз!

Кроки реалізації:

- 1) Як завжди починаємо з того, що обираємо прапор.
- 2) У графічному редакторі GIMP додаємо напис зеленого кольору на зображення.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		102

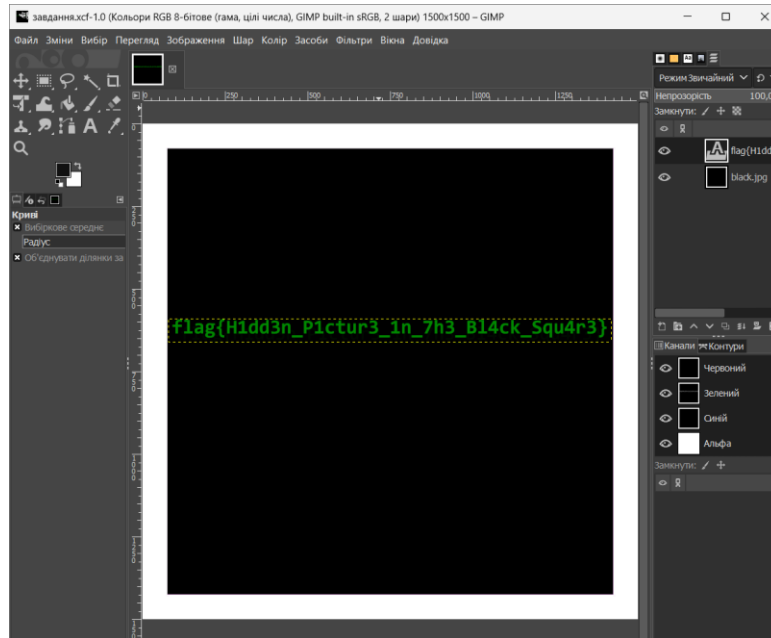


Рисунок 3.35 – Результат додавання напису на картину

- 3) Далі обираємо Колір Криві і зелений канал. Потім зміщуємо криву доти, поки наш напис повністю не зникне. Зберігаємо картинку.

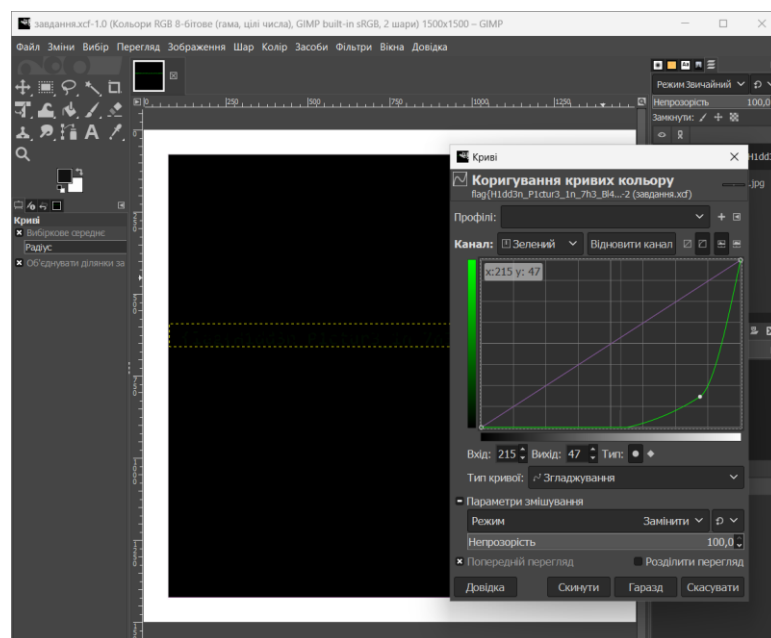


Рисунок 3.36 – Результат зміни налаштувань кривих

Інструменти для рішення завдання: aperisolve.fr, графічний редактор GIMP

Тестування та перевірка:

Є два шляхи перевірки цього завдання.

Перший дуже простий, за умови, що вам відомий інструмент aperisolve.fr.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		103

Ми просто завантажуємо картинку на сайт і відразу бачимо безліч копій зображення зі зміненими значеннями RGB каналів. Саме тому знайти прихований прапор досить легко.

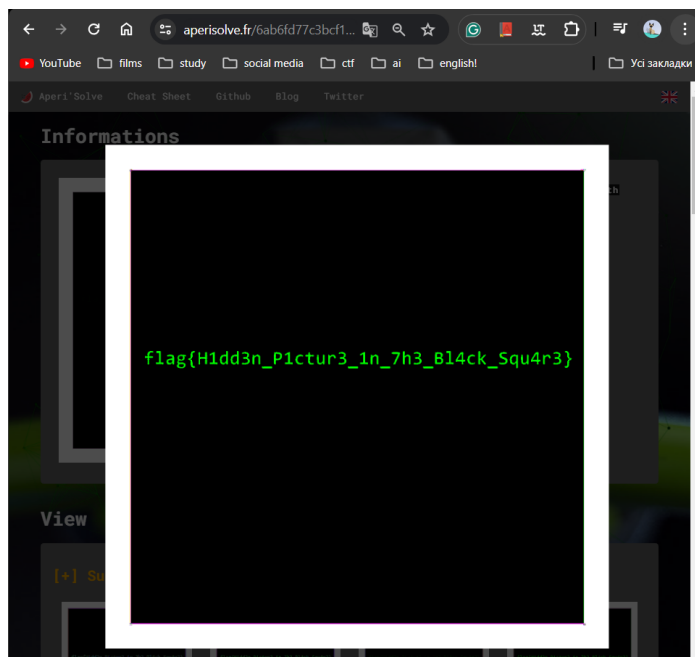


Рисунок 3.37 – Отриманий прапор

Інший варіант, це власноруч у будь-якому графічному редакторі змінювати значення всіх по черзі каналів, щоб віднайти прихований прапор.

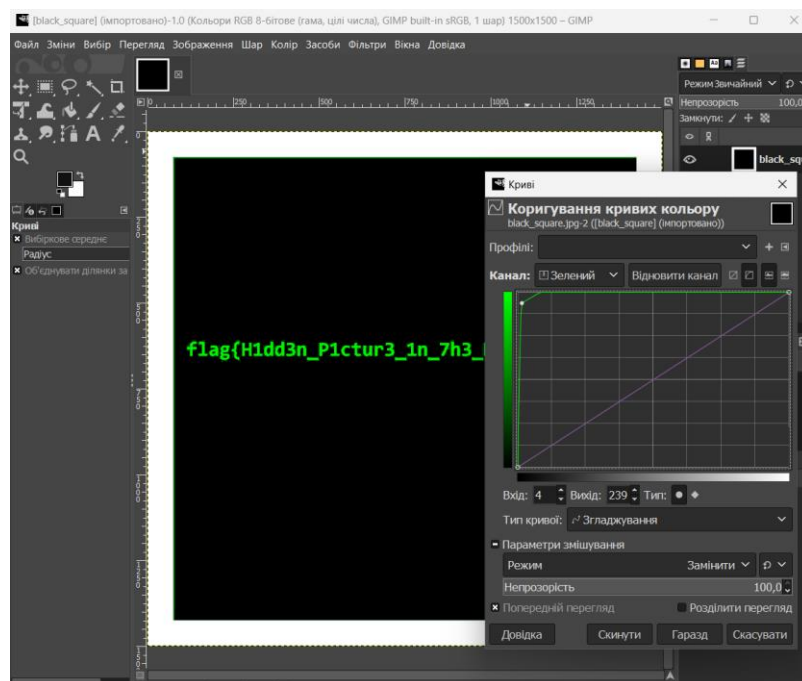


Рисунок 3.38 – Результат зміни значень кривих

Вихідні дані: flag{H1dd3n_P1ctur3_1n_7h3_Bl4ck_Squ4r3}

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				104
Змн.	Арк.	№ докум.	Підпис	Дата		

3.4 Практична реалізація стеганографічних завдань середнього та складного рівнів

3.4.1 «Mona Lisa»

Мета реалізації:

Це завдання зосереджене на пошуку незвичної картини серед двох тисяч копій. Особливістю картини є напис, який містить зашифрований прапор.

Учасникам потрібно ретельно дослідити архів картин, розглядаючи будь-які відмінності, наприклад, розмір зображення. Уважність та креативність у такому випадку ключ до успіху. Окрім цього, варто відшукати шифр, за допомогою якого цей прапор був зашифрований.

Використані техніки: символний шифр Mary Stuart Code, стеганографія зображення

Мови програмування/інструменти: мова програмування Python, графічний редактор GIMP, онлайн ресурс dcode.fr, WinRAR

Вхідні дані:

- **Назва завдання:** «Mona Lisa»
- **Опис до завдання:**

«Мона Ліза» є однією з найвідоміших картин у світі, створеною Леонардо да Вінчі. Через свою унікальність та значення, вона стала об'єктом багатьох копій та репродукцій.

Точну кількість копій та репродукцій «Джоконди» визначити неможливо. Їх існує безліч, адже протягом століть картину копіювали численні художники, як за життя да Вінчі, так і після його смерті. Деякі з них створювалися з метою навчання та вивчення техніки майстра, інші - як комерційні репродукції, а треті - як фальшивки для обману колекціонерів.

Серед безлічі картин знайдіть ту, яка приховує у собі особливу деталь.

- **Складність:** середній рівень
- **Формат прапора:** після виконання завдання ви отримаєте прапор формату flag{Слово1Слово2Слово3}

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		105

– **Файли для розв'язання завдання:** art_collection_of_Mona_Lisa.zip

– **Підказки:**

Підказки №1. Цікаво, в якому році Леонардо да Вінчі почав писати цю картину? (натяк на номер картини, яка містить прапор)

Підказки №2. Чи відомо Вам щось про Марія І Стюарт? (натяк на шифр Mary Stuart Code)

Кроки реалізації:

- 1) Обираємо рядок для створення прапору.
- 2) Використовуючи сайт dcode.fr, зашифруємо прапор.
- 3) Робимо кілька копій оригінального зображення.
- 4) За допомогою програми GIMP додаємо прапор на одне із зображень.
- 5) Змінемо розміри картинок (стиснемо їх) за допомогою онлайн ресурсу [Optimizilla](https://www.optimizilla.com/). Це необхідний крок.
- 6) Тепер у нас є три картинки різного розміру: 44 Кб, 42 Кб та 40 Кб. Далі нам потрібно оновити дату створення/зміни зображень, щоб вони всі мали однаковий час. Для цього напишемо скрипт time_changer.py та перевіримо його результат роботи.

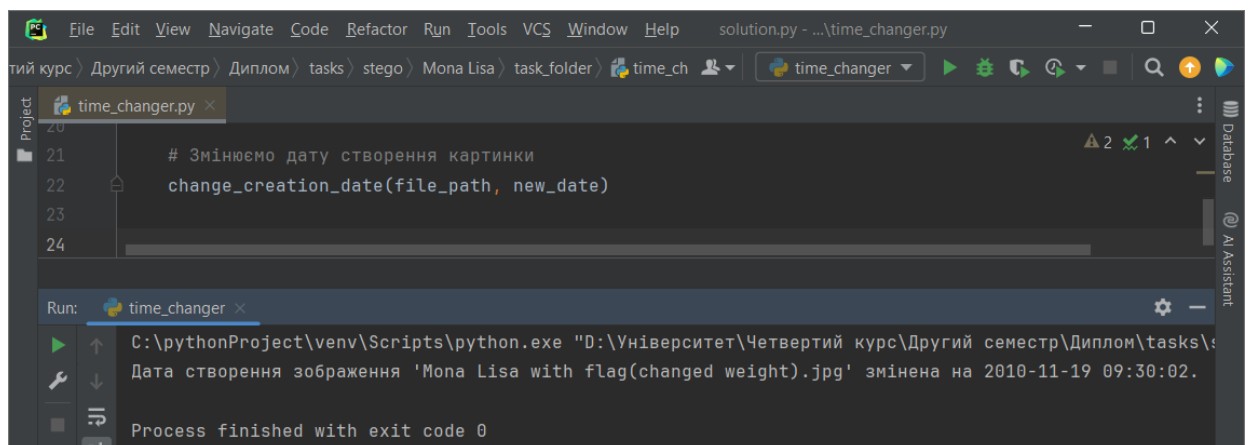


Рисунок 3.39 – Результат виконання коду

- 7) Наступний крок – це створення кількох тисяч копій зображення для того, щоб серед них приховати картину із прапором. Для цього теж використаємо мову програмування Python та напишемо скрипт сору_images_change_time.py, який поєднаємо з кодом time_changer.py.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		106

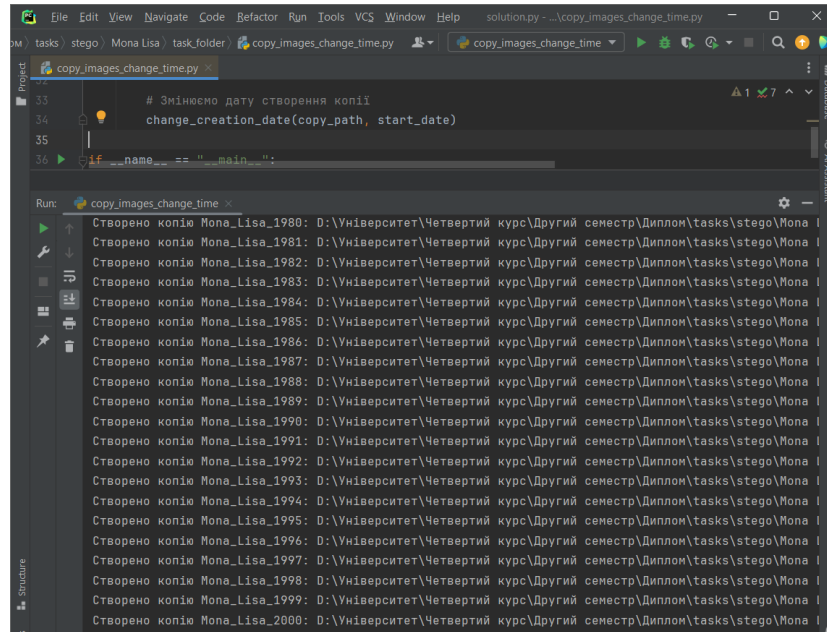


Рисунок 3.40 – Результат виконання коду

8) Замінюємо 1503 картинку на зображення з прапором. Число 1503 вибрано не випадково, адже це рік початку написання «Мона Лізи».

9) Створюємо zip-архів art_collection_of_Mona_Lisa.zip

Інструменти для рішення завдання: WinRAR, dcode.fr

Тестування та перевірка:

- 1) Пропустимо базові кроки, такі як розархівування архіву з картинками.
- 2) Найлегший спосіб знайти серед такої кількості картин потрібну нам – це відсортувати зображення за розміром.

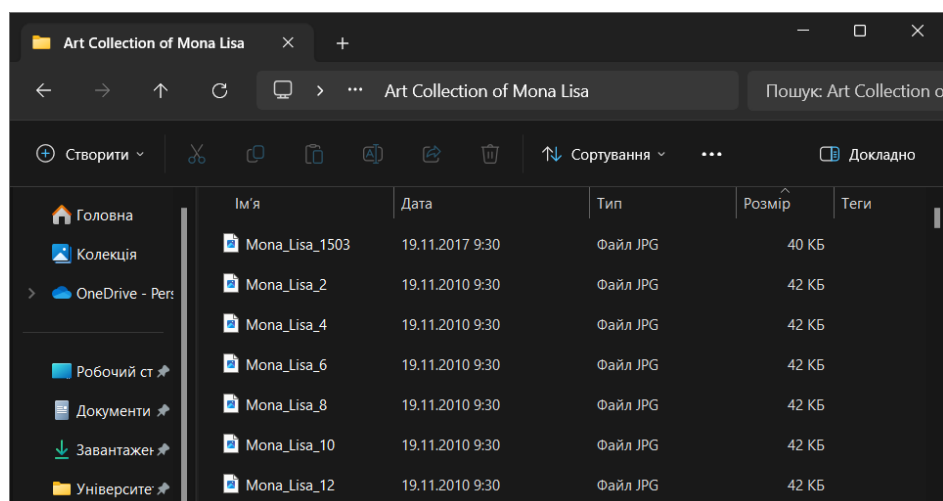


Рисунок 3.41 – Вміст архіву

Картина з найменшим розміром і є тією, яку ми шукаємо.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				107
Змн.	Арк.	№ докум.	Підпис	Дата		



Рисунок 3.42 – Зображення із зашифрованим прапором

- 3) Бачимо досить незвичний шифр. Спробуємо знайти його серед списку символічних шифрів на сайті dcode.fr.
- 4) Після цього дешифруємо прапор за допомогою шифру Mary Stuart Code.

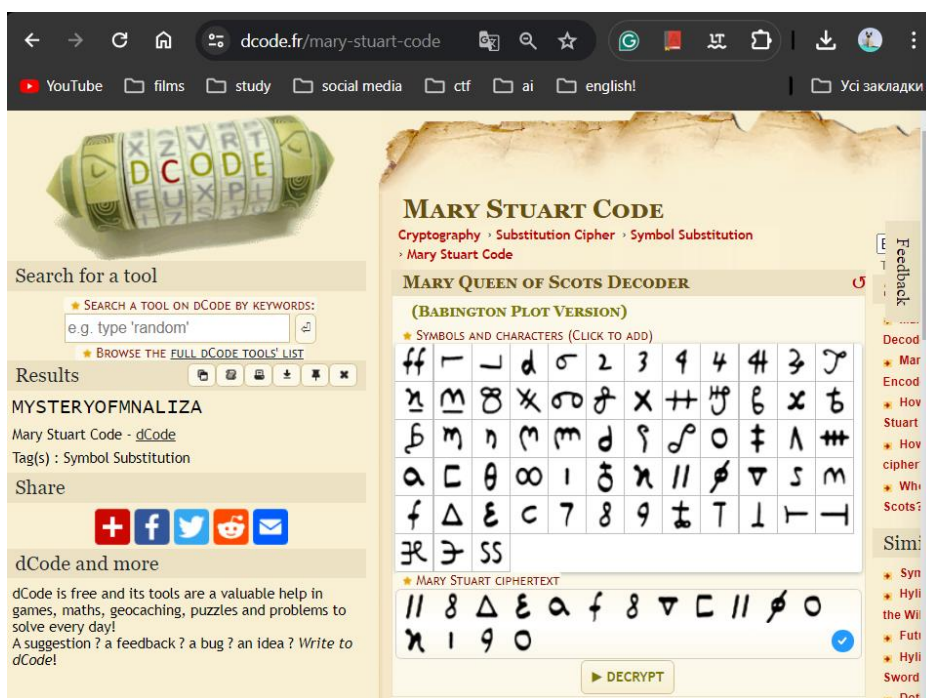


Рисунок 3.43 – Результат розшифрування прапора

Вихідні дані: flag{MysteryOfMonaLiza}

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				108
Змн.	Арк.	№ докум.	Підпис	Дата		

3.4.2 «Silent Nights, Ode to Winter»

Мета реалізації:

Завдання зосереджене на аналізі тексту, вивченні нових технік та інструментів для викриття прихованого повідомлення. Окрім того, учасникам варто ще ознайомитися з досі невідомими утилітами для роботи із стеганографією зображення та методами брутфорсу.

Використані техніки: стеганографія тексту та зображення, метод пробілів (white steg), брутфорс паролів

Мови програмування/інструменти: stegsnow, steghide

Вхідні дані:

- **Назва завдання:** Silent Nights, Ode to Winter
- **Опис до завдання:**

У затишному куточку музею, де пил століть лягав на безліч картин, висіла одна, немов забута. Її сюжет простий: засніжений пейзаж з деревами, вкритими снігом. Картина не вирізнялась ані відомим автором, ані вишуканою технікою, але чомусь притягувала погляди.

Проходячи повз залу з картиною, одна з відвідувачок несподівано зупинилася, зачаровано прикипівши поглядом до полотна, яке здалося їй знайомим, хоча вона не могла зрозуміти чому. Підійшовши ближче, дівчина почала уважно розглядати деталі, звернувши увагу не лише на мальовничі пейзажі, а й на химерні візерунки, які виднілися крізь сніговий покрив, і помітила, що вони складали літери.

- **Складність:** середній рівень
- **Формат прапора:** після виконання завдання ви отримаєте прапор формату flag{Слово1_Слово2}
- **Файли для розв'язання завдання:** winter_painting.jpg

Кроки реалізації:

- 1) Першим кроком має бути написання примітивного вірша і обрання рядка для прапора.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		109

- 2) Далі використаємо утиліту stegsnow для приховання прапорця серед рядків поезії.

```
kali-linux-2023.2-virtualbox-amd64 [Запущено] - Oracle VM VirtualBox
Файл  Машина  Перегляд  Введення  Пристрої  Довідка
File  Actions  Edit  View  Help
(kali@kali)-[/media/sf_/tasks/stego/Silent Nights, Ode to Winter]
$ stegsnow -C -m "flag{S3cr3t5_0f_N4tur3_4r3_H1dd3n_Und3r_Th3_Sn0w}" -p "000000" original-poetry.txt winter_poetry.txt
Compressed by 4.59%
Message exceeded available space by approximately 3.89%.
An extra 1 lines were added.
(kali@kali)-[/media/sf_/tasks/stego/Silent Nights, Ode to Winter]
$
```

Рисунок 3.44 – Результат приховування прапора

- 3) Новостворений текстовий файл winter_poetry.txt заховуємо у картині. Для цього використаємо стеганографічну програму steghide.

```
kali-linux-2023.2-virtualbox-amd64 [Запущено] - Oracle VM VirtualBox
Файл  Машина  Перегляд  Введення  Пристрої  Довідка
File  Actions  Edit  View  Help
(kali@kali)-[/media/sf_/tasks/stego/Silent Nights, Ode to Winter]
$ steghide embed -ef winter_poetry.txt -cf winter_painting.jpg
Enter passphrase:
Re-Enter passphrase:
embedding "winter_poetry.txt" in "winter_painting.jpg" ... done
(kali@kali)-[/media/sf_/tasks/stego/Silent Nights, Ode to Winter]
$
```

Рисунок 3.45 – Результат приховування текстового файлу в зображенні

Інструменти для рішення завдання: stegseek або steghide, stegsnow, stegsnowbruteforcer.

Тестування та перевірка:

- 1) Використаємо stegseek для отримання прихованого текстового файлу та переглянемо його вміст.

```

kali-linux-2023.2-virtualbox-amd64 [Запущено] - Oracle VM VirtualBox
Файл Машина Перегляд Введення Пристрої Довідка
File Actions Edit View Help

(kali@kali)-[/media/sf_/tasks/stego/Silent Nights, Ode to Winter]
$ stegseek -sf winter_painting.jpg

StegSeek 0.6 - https://github.com/RickdeJager/StegSeek

[i] Found passphrase: "snowflake"
[i] Original filename: "winter_poetry.txt".
[i] Extracting to "winter_painting.jpg.out".

(kali@kali)-[/media/sf_/tasks/stego/Silent Nights, Ode to Winter]
$ cat winter_painting.jpg.out
In winter's hush, the snowflakes dance,
A silent waltz, a fleeting glance.
They blanket fields in purest white,
A canvas for the dawn's first light.

But as the days grow longer, see,
The thaw begins, the streams run free.
The icy grip of winter wanes,
As springtime whispers in soft refrains.

The buds emerge, the colors bloom,
The world awakes from winter's gloom.
A symphony of life's rebirth,
As nature paints the earth with mirth.

Yet in the whispers of the breeze,
A hint of winter's memories.
For seasons change, but memories stay,
In snow and spring, in night and day.

(kali@kali)-[/media/sf_/tasks/stego/Silent Nights, Ode to Winter]
$

```

Рисунок 3.46 – Результат отримання прихованого текстового файлу

- 2) Далі за допомогою утиліти stegsnow та відомого нам паролю отримаємо прихований прапор.

```

kali-linux-2023.2-virtualbox-amd64 [Запущено] - Oracle VM VirtualBox
Файл Машина Перегляд Введення Пристрої Довідка
File Actions Edit View Help

(kali@kali)-[/media/sf_/tasks/stego/Silent Nights, Ode to Winter]
$ stegsnow -C -p "000000" winter_painting.jpg.out

flag{S3cr3t5_0f_N4tur3_4r3_H1dd3n_Und3r_Th3_Sn0w}

(kali@kali)-[/media/sf_/tasks/stego/Silent Nights, Ode to Winter]
$

```

Рисунок 3.47 – Результат отримання прихованого прапора

- 3) Учасникам для того, щоб отриманий прихований прапор в тексті, доведеться скористатися будь-якою відомою їм утилітою для брутфорсу, наприклад, stegsnowbruteforcer.

```

kali-linux-2023.2-virtualbox-amd64 [Запущено] - Oracle VM VirtualBox
Файл Машина Перегляд Введення Пристрої Довідка
File Actions Edit View Help
(kali@kali)-[/media/.../tasks/stego/Silent Nights, Ode to Winter/StegnowBruteForcer]
$ python stegnowbruteforcer.py --file winter_poetry.txt --wordlist /usr/share/wordlists/rockyou.txt --keyword flag! --output results.txt

*****
* Stegnow Password Brute-Force Attack *
* made by Assa *
* https://github.com/Assa228/StegnowBruteForcer *
*****

0%|          | 11/14344392 [01:09<32727:27:27, 8.21s/it]
Keyword 'flag{' found with the password: 000000
Here is the decoded message:
flag{S3cr3t5_0f_N4tur3_4r3_H1dd3n_Und3r_Th3_Sn0w}
0%|          | 446/14344392 [01:13<31:57:50, 124.65it/s]

```

Рисунок 3.48 – Результат брутфорсу файлу

Вихідні дані: flag{S3cr3t5_0f_N4tur3_4r3_H1dd3n_Und3r_Th3_Sn0w}

3.4.3 «Art in the Digital Age»

Мета реалізації:

Завдання полягає в приховуванні аудіофайлу шляхом перетворення його на зображення формату PNG. Учасники повинні логічно підходити до виконання завдання та дослідити зображення, щоб знайти спосіб відновити аудіо з прапором.

Використані техніки: стеганографія зображення та аудіо

Мови програмування/інструменти: сайт для створення аудіофайлів, наприклад, <https://play.ht/>, ресурс для реверсу музики та сайт <https://directmusic.me/wav2png/>.

Вхідні дані:

- **Назва завдання:** «Art in the Digital Age»
- **Опис до завдання:**

Нещодавно в нашому музеї з'явилася неймовірна робота, яка безперервно захоплює увагу всіх відвідувачів. Це полотно, яке, здавалося б, можна не лише побачити, але й почути. Майстерність художника полягала в тому, що він вмщав в кожну краплинку фарби не тільки красу образу, але й звук.

Погляньте глибше, щоб відчути шепіт картини.

- **Складність:** середній рівень
- **Формат прапора:** після виконання завдання ви отримаєте прапор формату flag{Слово1_Слово2}

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				112
Змн.	Арк.	№ докум.	Підпис	Дата		

– **Файли для розв'язання завдання:** sunset_painting.png

Кроки реалізації:

- 1) Обираємо рядок для створення прапорця.
- 2) Знаходимо сайт, який зміг би озвучити наш прапор, наприклад, play.ht.

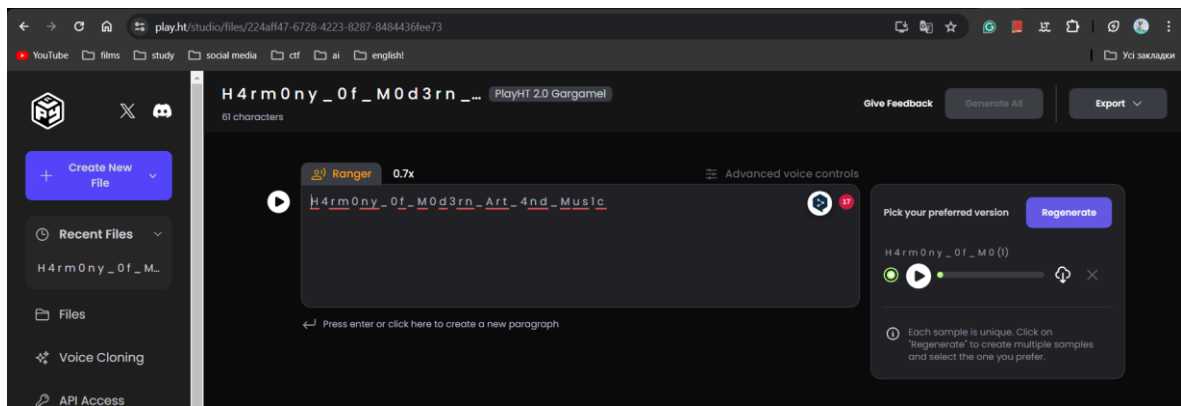


Рисунок 3.49 – Створення звукового файлу із прапорцем

- 3) Далі нам потрібен інструмент для відтворення звуку у зворотньому порядку, наприклад, audioalter.com.
- 4) Ще одним важливим кроком є перетворення звуку в картинку. Використаємо сайт directmusic.me для цього.

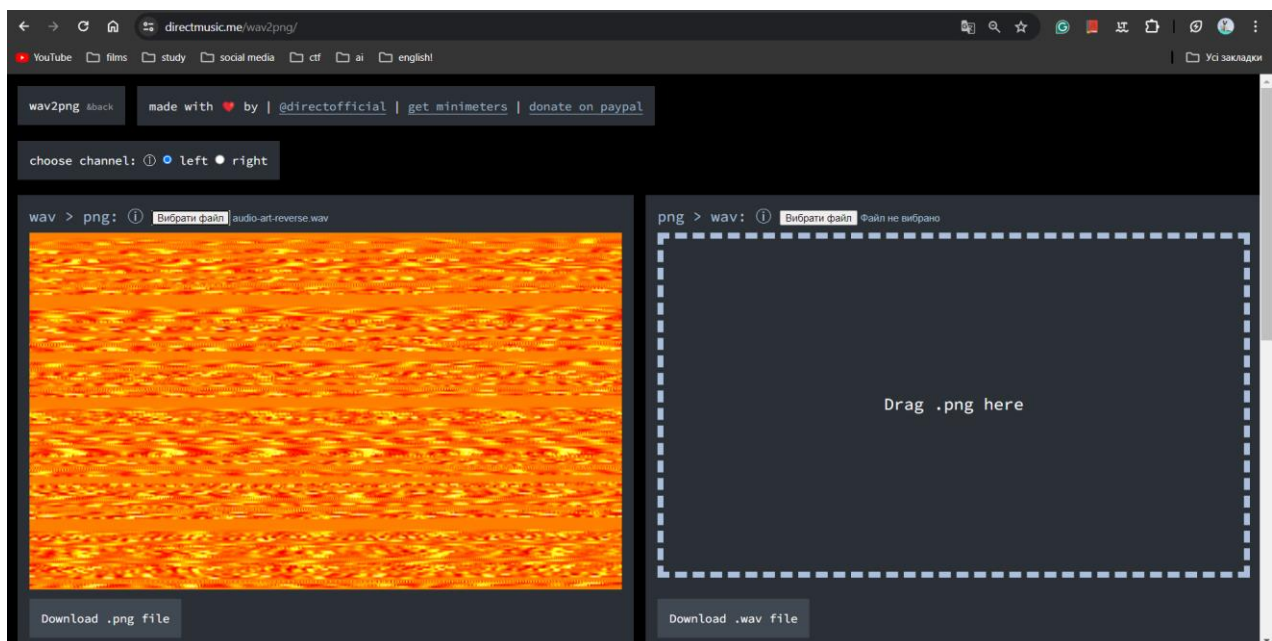


Рисунок 3.50 – Результат перетворення звуку в зображення

- 5) Заархівуємо отримане зображення, а потім об'єднаємо JPG-зображення пейзажу і ZIP-архів у новий файл з ім'ям sunset_painting.jpg.

```

C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.22631.3593]
(c) Корпорація Майкрософт. Усі права захищені.

D:\Університет\Четвертий курс\Другий семестр\Диплом\tasks\stego\Art in the Digital Age>copy /b sunset-oil-painting(original).jpg + sunset.zip sunset_painting.jpg
sunset-oil-painting(original).jpg
Overwrite sunset_painting.jpg? (Yes/No/All): Yes
sunset.zip
1 file(s) copied.

D:\Університет\Четвертий курс\Другий семестр\Диплом\tasks\stego\Art in the Digital Age>

```

Рисунок 3.51 – Результат об’єднання файлів

Інструменти для рішення завдання: binwalk чи aperisolve.fr для отримання прихованої картинки, directmusic.me для перетворення зображення назад у звук, інструмент для реверсу звуку (audioalter.com).

Тестування та перевірка:

1) Використаємо aperisolve.fr та binwalk для отримання зображення.

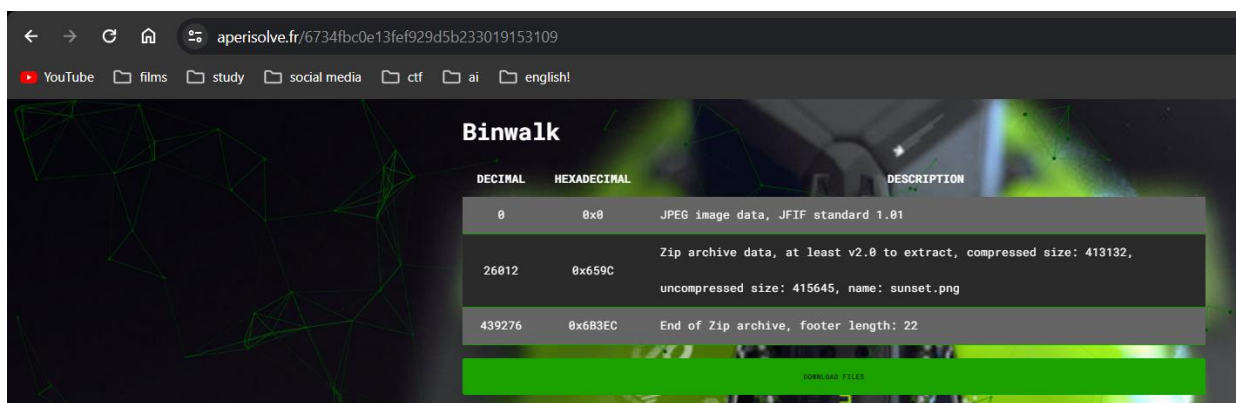


Рисунок 3.52 – Результат перегляду вмісту зображення

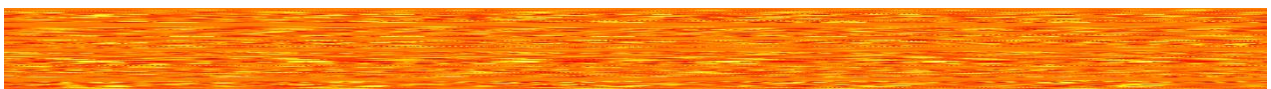


Рисунок 3.53 – Отримане зображення

2) Далі це зображення варто перетворити в звук, відтворити його у зворотньому порядку для того, щоб почути літери, які складають прапор.

Вихідні дані: flag{H4rmony_of_M0d3rn_Art_4nd_Mus1c}

3.4.4 «Trojan Horse»

Мета реалізації:

Завдання полягає в тому, що учасникам потрібно буде виявити прихований у зображенні архів. Далі варто логічно помислити і знайти спосіб, у який вони об’єднують всі шматочки QR-коду. Можливо, для виконання завдання слід використати

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		114

спеціальні програми для з'єднання фрагментів зображення і отримання інформації, яка прихована всередині.

Використані техніки: стеганографія зображення, QR-код

Мови програмування/інструменти: мова програмування Python, онлайн ресурс [the-qrcode-generator](https://the-qrcode-generator.com/), WinRAR.

Вхідні дані:

- ***Назва завдання:*** «Trojan Horse»
- ***Опис до завдання:***

Троянський кінь - це міфічний об'єкт з грецької міфології, який став відомий завдяки Одиссеї Гомера. Ця історія розповідає про те, як греки, не здатні проникнути в місто Трої, побудували величезного дерев'яного коня і приховали всередині воїнів.

Цікаво, що приховує ця фігурка Троянського коня.

- ***Складність:*** середній рівень
- ***Формат прапора:*** після виконання завдання ви отримаєте прапор формату flag{Слово1_Слово2}
- ***Файли для розв'язання завдання:*** зображення trojan_horse.jpg

Кроки реалізації:

- 1) Як і завжди почати варто з обрання рядка, який стане прапором.
- 2) Далі за допомогою [онлайн-сервісів](#) створимо з нього QR-код (Quick Response code).
- 3) Наступний крок – це написання коду, який розкладе згенерований QR-код на шматочки розміру 25x25 пікселів. Всього вийшло 400 таких картинок.
- 4) Створимо архів та за допомогою командного рядка помістимо його у зображення trojan_horse.jpg

Інструменти для рішення завдання: мова програмування Python, онлайн ресурс [the-qrcode-generator](https://the-qrcode-generator.com/), WinRAR.

Тестування та перевірка:

- 1) Витягнувши із зображення архів та розархівувавши його, ми маємо написати код, який об'єднає всі шматки QR-коду.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		115

```

mystery_pieces\solve_qr-code.py x
1  from PIL import Image
2
3  # Створення нового зображення розміром 500x500
4  combined_image = Image.new( mode: 'RGB', size: (500, 500))
5
6  # Перевірка, чи є всі 400 шматочків картинки

```

Run: solve_qr-code (1) x

C:\pythonProject\venv\Scripts\python.exe "D:\Університет\Четвертий курс\Другий семес
06'єднане зображення збережено як 'combined_image.png'.

Process finished with exit code 0

Рисунок 3.54 – Результат виконання коду



Рисунок 3.55 – Отриманий QR - код

- 2) Після цього за допомогою спеціальних програм перевіримо роботу QR-коду.

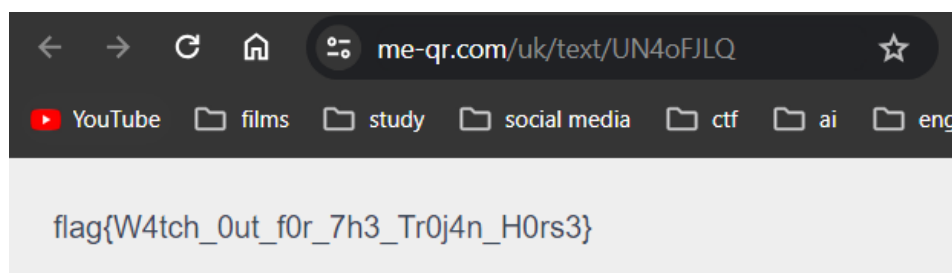


Рисунок 3.56 – Результат сканування QR - коду

Вихідні дані: flag{W4tch_Out_f0r_7h3_Tr0j4n_H0rs3}

3.4.5 «Museum Advertising»

Мета реалізації:

Завдання зосереджене на використанні різних інструментів та містить певну кількість шарів. Тож учасникам потрібно буде крок за кроком відкривати новий шар та знаходити приховані частини прапора.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		116

Використані техніки: стеганографія відео та зображення, зміна метаданих, кодування Base32, Base16.

Мови програмування/інструменти: Microsoft Clipchamp, GIMP, exiftool, zip, steghide, сайт для завантаження зображень <https://ibb.co/>, Outguess, ресурс для створення QR-кодів <https://me-qr.com/qr-code-generator>, CyberChef, Openstego.

Вхідні дані:

- **Назва завдання:** «Museum Advertising»
- **Опис до завдання:**

Створення реклами для локального музею вкрай важливе з точки зору залучення нових відвідувачів та збільшення свідомості про його існування. Тож наш музей також вирішив змонтувати свій рекламний ролик, щоб привернути увагу людей до чудових колекцій та цікавих виставок.

Перегляньте це відео та поділіться враженнями. Гарного перегляду!

- **Складність:** складний рівень
- **Формат прапора:** після виконання завдання ви отримаєте прапор формату flag{Слово1_Слово2}
- **Файли для розв'язання завдання:** museum advertising.mp4
- **Підказки:**

Підказка №1. Автор рекламного відео відданий фанат Гаррі Поттера. А чи знаєте ви на скільки насправді частин Лорд Волдеморт розділив свою душу, щоб створити горокракси? (7 горокраксів).

Кроки реалізації:

- 1) Обираємо дуже довгий рядок для створення прапора. Далі розділяємо його на 7 частин, кожна з яких буде деь прихована.
- 2) Це завдання матиме багато нашарувань. Тому варто працювати у зворотньому напрямку.

Для приховання сьомої частини створимо текстовий файл, який заховаємо у картині за допомогою утиліти OpenStego. Використаний пароль qwerty12345.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		117

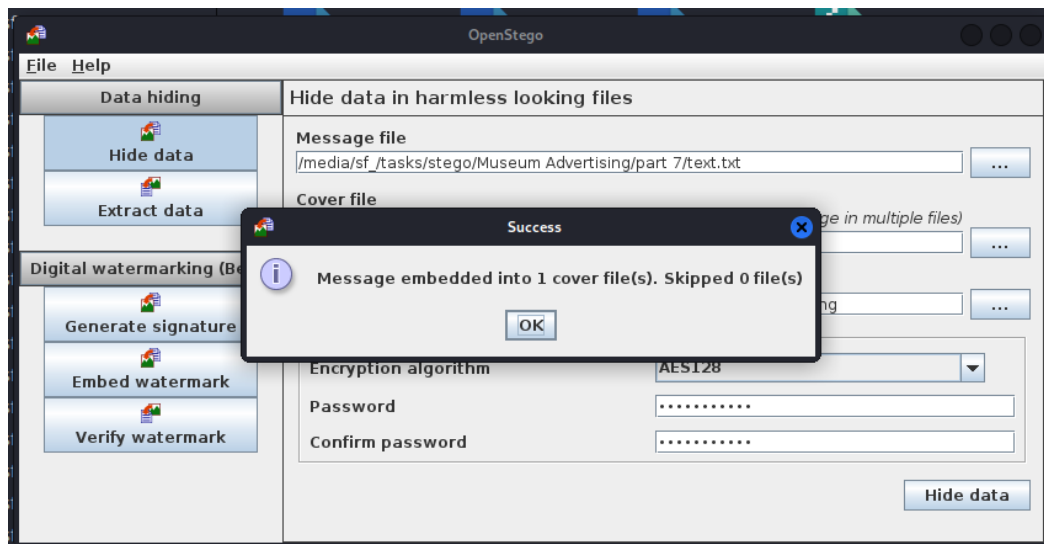


Рисунок 3.57 – Приховання файлу за допомогою утиліти OpenStego

- 3) Завантажимо картину на спеціальний ресурс, створений для того, щоб злегкістю ділитися зображеннями. Посилання на картинку закодуємо Base32 за допомогою онлайн-інструменту CyberChef. Результат кодування посилання:

NB2HI4DTNIXS62LCMIXGG3ZPKQYDGNCYGE2A====

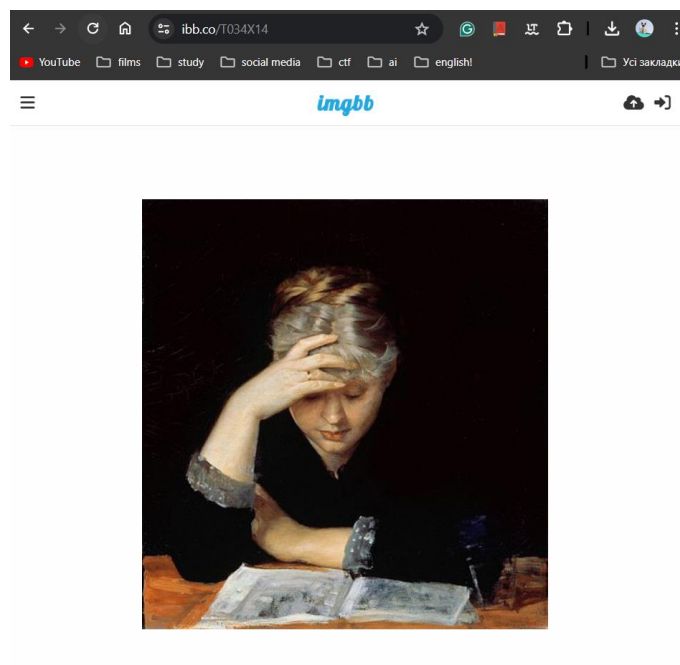


Рисунок 3.58 – Завантажена картина

- 4) Закодоване посилання буде підказкою залишеною в текстовому файлі, але спершу його треба створити. Переходимо на сайт для генерації QR-кодів.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				118
Змн.	Арк.	№ докум.	Підпис	Дата		

Наш QR-код буде містити шосту частину прапора і підказку.
Було б занадто просто залишити QR-код у форматі зображення, тому знову використаємо CyberChef, але цього разу HEX кодування файлу.

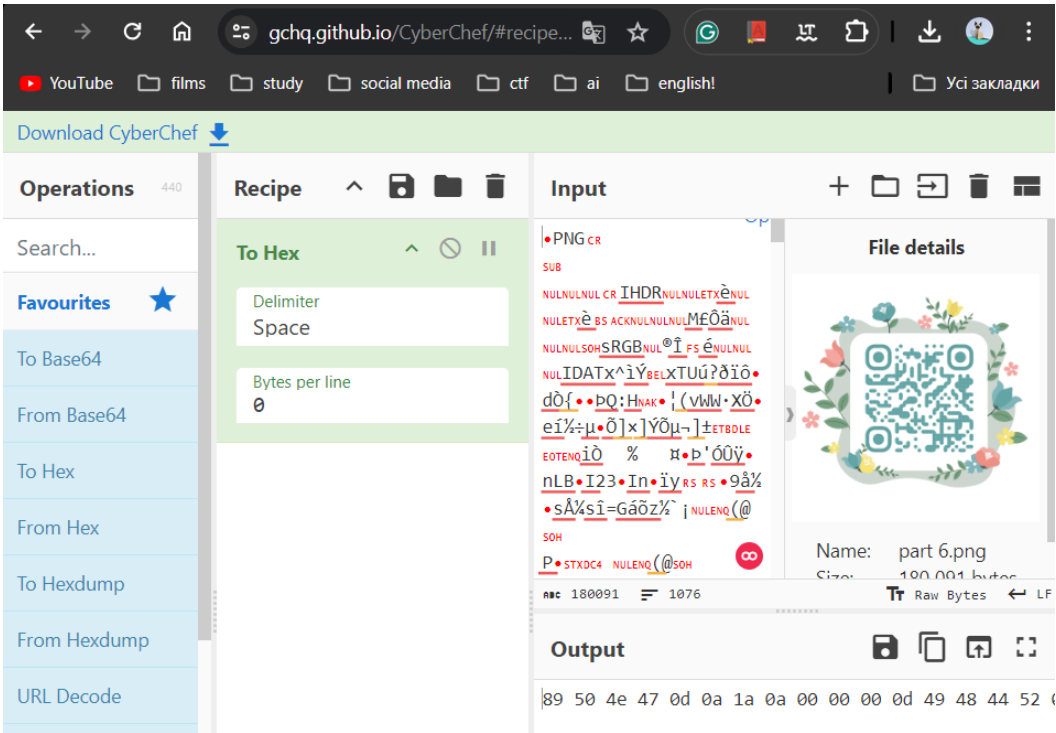


Рисунок 3.59 – Результат кодування файлу

Отриманий код, підказку та четверту частину прапора запишемо в текстовий файл.

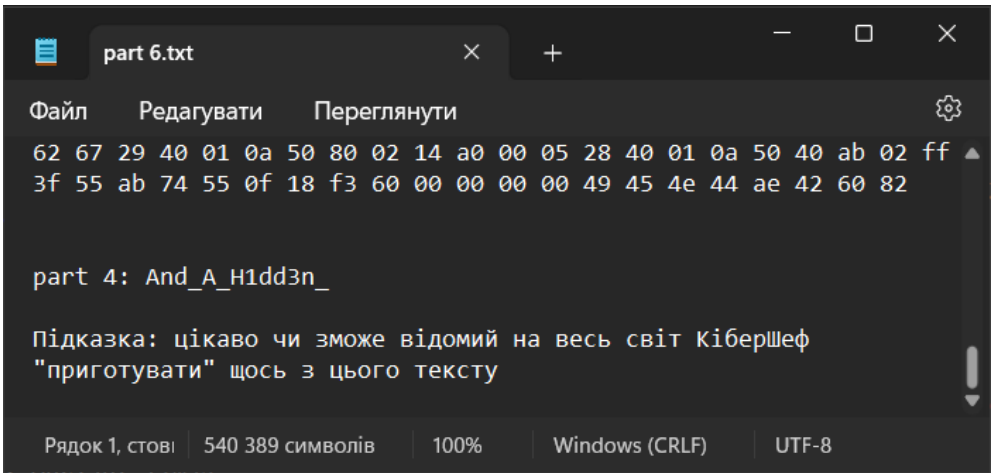


Рисунок 3.60 – Вміст файлу з прапором

- 5) П'яту частину прапора та закодоване посилання додамо до ще одного текстового файлу. Його ми приховуємо в зображенні за допомогою утиліти Outguess.

```

kali@kali:~/media/.../tasks/stego/Museum Advertising/part 5$ outguess -k "To be, or not to be, that is the question" -d hidden.txt "part 5.jpg" "part 5 with flag.jpg"
Reading part 5.jpg...
JPEG compression quality set to 75
Extracting usable bits: 58153 bits
Correctable message size: 48051 bits, 82.63%
Encoded 'hidden.txt': 1296 bits, 162 bytes
Finding best embedding...
0: 700(52.7%)[54.0%], bias 735(1.05), saved: -6, total: 1.20%
1: 671(50.5%)[51.8%], bias 680(1.01), saved: -2, total: 1.15%
2: 676(50.9%)[52.2%], bias 651(0.96), saved: -3, total: 1.16%
3: 671(50.5%)[51.8%], bias 644(0.96), saved: -2, total: 1.15%
4: 657(49.5%)[50.7%], bias 652(0.99), saved: -1, total: 1.13%
6: 647(48.7%)[49.9%], bias 569(0.88), saved: 0, total: 1.11%
168: 636(47.9%)[49.1%], bias 569(0.89), saved: 1, total: 1.09%
168, 1205: Embedding data: 1296 in 58153
Bits embedded: 1328, changed: 636(47.9%)[49.1%], bias: 569, tot: 58038, skip: 56710
Foiling statistics: corrections: 214, failed: 0, offset: 57.747899 +- 113.540795
Total bits changed: 1205 (change 636 + bias 569)
Storing bitmap into data...
Writing part 5 with flag.jpg...

```

Рисунок 3.61 – Результат приховання файлу в зображенні

- 6) Отримані раніше текстовий файл та зображення помістимо у захищений паролем архів. Далі цей архів вбудуємо в зображення, тим самим приховавши його.

```

kali@kali:~/media/.../tasks/stego/Museum Advertising/part 4$ zip -r -P thisispassword archive.zip files
adding: files/ (stored 0%)
adding: files/image.jpg (deflated 3%)
adding: files/text.txt (deflated 59%)

```

Рисунок 3.62 – Результат архівування файлів

```

C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.22631.3593]
(c) Корпорація Майкрософт. Усі права захищені.

D:\Університет\Четвертий курс\Другий семестр\Диплом\tasks\stego\Museum Advertising\part 4>copy /b "part 4.jpg" + archive.zip "part 4 with archive.jpg"
part 4.jpg
archive.zip
1 file(s) copied.

D:\Університет\Четвертий курс\Другий семестр\Диплом\tasks\stego\Museum Advertising\part 4>

```

Рисунок 3.63 – Результат вбудування архіву в картинку

- 7) Перейдемо до наступного етапу. Тут нам потрібно відредагувати зображення і додати на нього напис з частиною прапора. Використаємо вже

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				120
Змн.	Арк.	№ докум.	Підпис	Дата		

знайомий редактор GIMP. Це зображення теж буде контейнером для попереднього файлу.

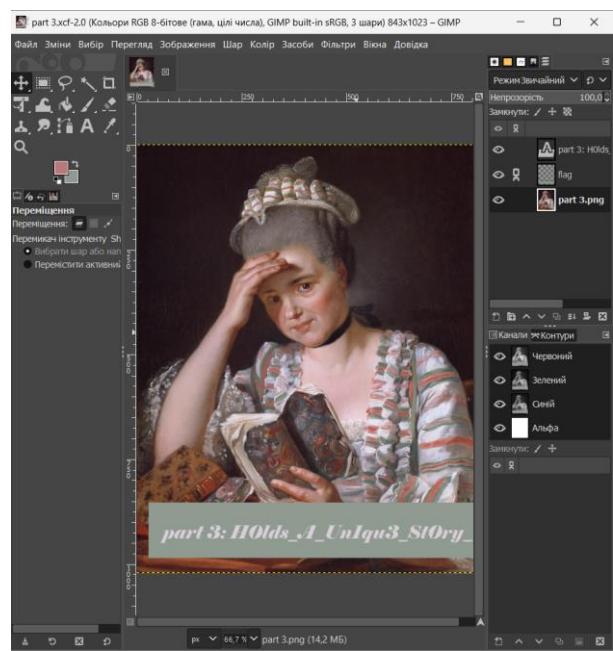


Рисунок 3.64 – Результат редагування зображення

Використаємо утиліту Steghide для приховання зображення image2. jpg. Потім заархівуємо отримане зображення.

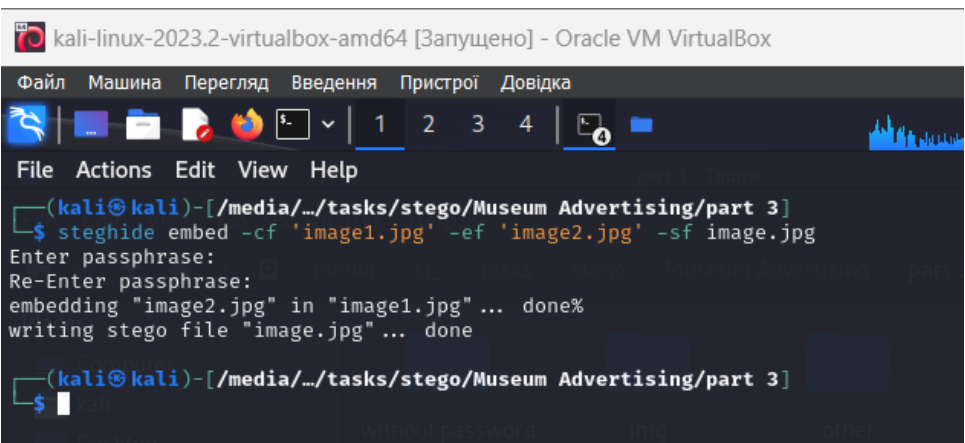


Рисунок 3.65 – Результат приховування файлів

- 8) Завантажуємо відео, яке міститиме кілька частин прапора та буде контейнером для архіву.
 - Одну з частин прапора приховуємо у самому відео. Для цього використовуємо відеоредактор Microsoft Clipchamp.

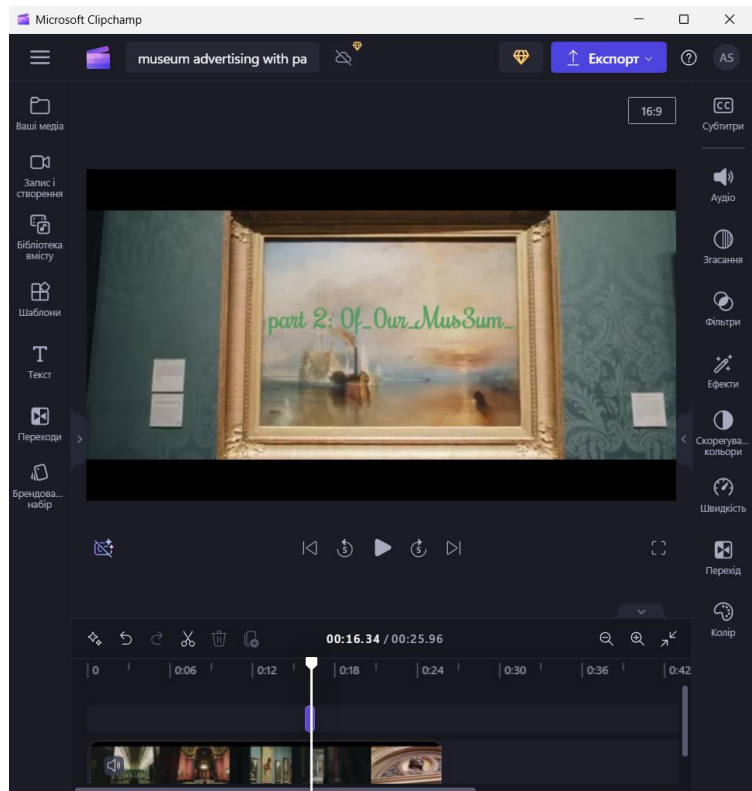


Рисунок 3.66 – Результат додавання прапора у відео

- Інший шматок буде приховано у метаданих відео, окрім цього додамо ще підказку.

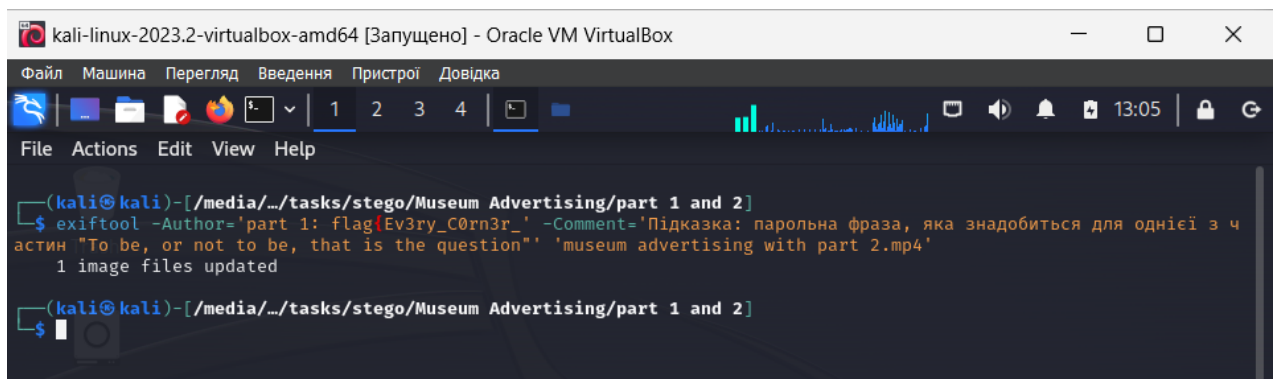


Рисунок 3.67 – Зміна метаданих відео

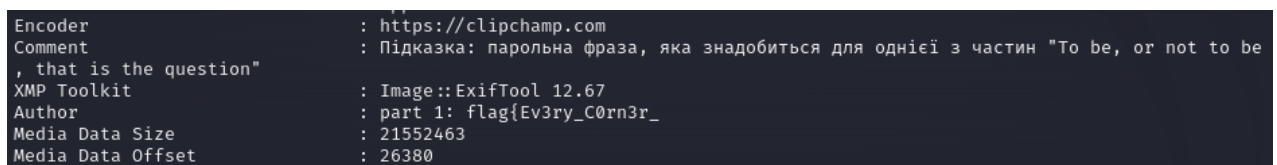


Рисунок 3.68 – Перевірка метаданих відео

- 9) Фінальний крок – це використати відео як контейнер для архіву.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				122
Змн.	Арк.	№ докум.	Підпис	Дата		

```

C:\Windows\System32\cmd.e x + v
Microsoft Windows [Version 10.0.22631.3593]
(c) Корпорація Майкрософт. Усі права захищені.

D:\Університет\Четвертий курс\Другий семестр\Диплом\tasks\stego\Museum Advertising\part 1 and 2>copy /b "video.mp4" + image.zip "video with archive.mp4"
video.mp4
image.zip
        1 file(s) copied.

D:\Університет\Четвертий курс\Другий семестр\Диплом\tasks\stego\Museum Advertising\part 1 and 2>

```

Рисунок 3.69 – Результат об'єднання файлів

Інструменти для рішення завдання: exiftool, unzip, steghide, stegseek, Out-guess, fcrackzip, CyberChef, Openstego.

Тестування та перевірка:

Етап тестування досить громісткий, тому опишемо лише важливі кроки.

- 1) Перші дві частини прапора заховані у відео: у матаданих та серед одного з кадрів. Об'єднаємо ці частини: `flag{Ev3ry_C0rn3r_of_0ur_Mus3um_`
- 2) Ще одна частина міститься на зображенні.

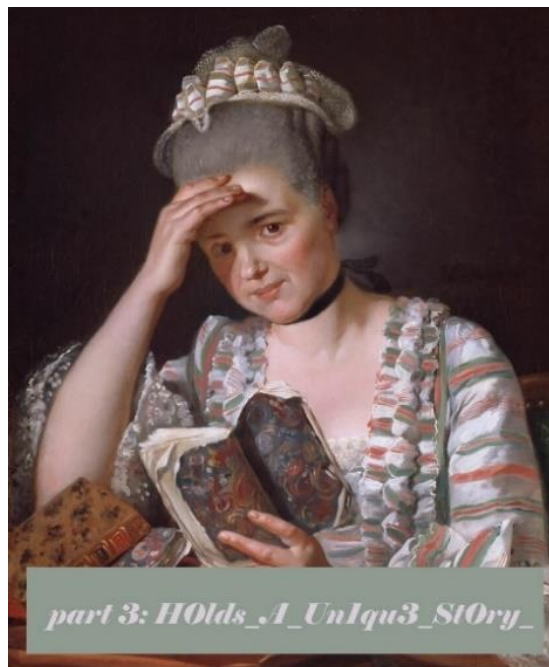


Рисунок 3.70 – Зображення з прапором

- 3) В цій картинці містилася ще одна, яку ви отримали за допомогою інструменту stegseek.

```

kali-linux-2023.2-virtualbox-amd64 [Запущено] - Oracle VM VirtualBox
Файл  Машина  Перегляд  Введення  Пристрої  Довідка
File  Actions  Edit  View  Help

(kali@kali)-[/media/.../tasks/stego/Museum Advertising/_museum advertising.mp4.extracted]
$ stegseek image.jpg rockyou.txt
StegSeek 0.6 - https://github.com/RickdeJager/StegSeek

[1] Found passphrase: "sunshine"

[i] Original filename: "image2.jpg".
[i] Extracting to "image.jpg.out".

(kali@kali)-[/media/.../tasks/stego/Museum Advertising/_museum advertising.mp4.extracted]
$

```

Рисунок 3.71 – Результат використання утиліти stegseek



Рисунок 3.72 – Отримане зображення

- 4) На цьому етапі виконуємо кілька базових операцій: перевірка та отримання вмісту зображення, розархівування архіву за допомогою fcrackzip.

```

kali-linux-2023.2-virtualbox-amd64 [Запущено] - Oracle VM VirtualBox
Файл  Машина  Перегляд  Введення  Пристрої  Довідка
File  Actions  Edit  View  Help

(kali@kali)-[/media/.../stego/Museum Advertising/_museum advertising.mp4.extracted/_image1.jpg.extracted]
$ ls
1E40C.zip  files

(kali@kali)-[/media/.../stego/Museum Advertising/_museum advertising.mp4.extracted/_image1.jpg.extracted]
$ file 1E40C.zip
1E40C.zip: Zip archive data, at least v1.0 to extract, compression method=store

(kali@kali)-[/media/.../stego/Museum Advertising/_museum advertising.mp4.extracted/_image1.jpg.extracted]
$ unzip 1E40C.zip
Archive:  1E40C.zip
[1E40C.zip] files/image.jpg password:
  skipping: files/image.jpg      incorrect password
  skipping: files/text.txt       incorrect password

(kali@kali)-[/media/.../stego/Museum Advertising/_museum advertising.mp4.extracted/_image1.jpg.extracted]
$ fcrackzip -D -p rockyou.txt -v -u 1E40C.zip

'files/' is not encrypted, skipping
found file 'files/image.jpg', (size cp/uc 59156/ 61067, flags 9, chk 91bd)
found file 'files/text.txt', (size cp/uc 221583/540468, flags 9, chk 8c5d)
checking pw budayl

PASSWORD FOUND!!!!: pw = thisispassword

(kali@kali)-[/media/.../stego/Museum Advertising/_museum advertising.mp4.extracted/_image1.jpg.extracted]
$

```

Рисунок 3.73 – Результат виконання базових операцій

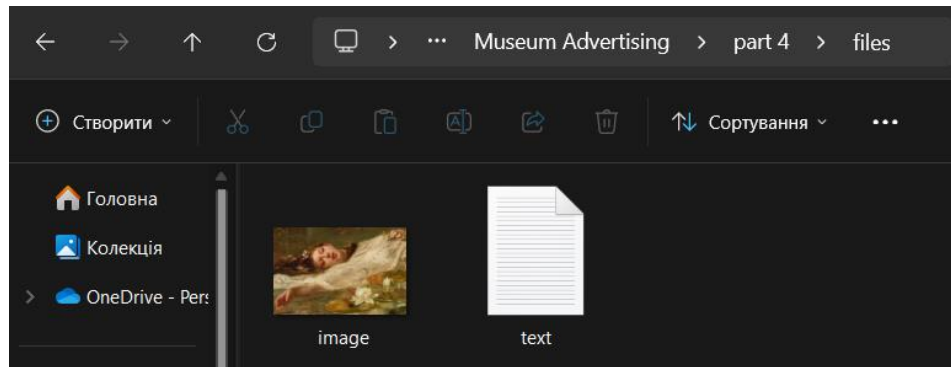


Рисунок 3.74 – Отримані зображення та текстовий файл

5) Переглядаючи отриманий текстовий файл знаходимо четветий шматочок.

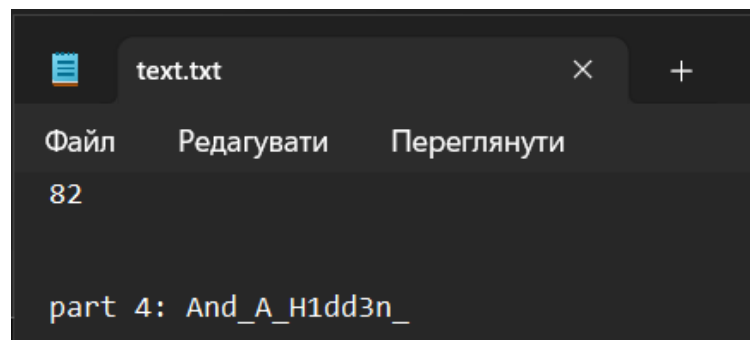


Рисунок 3.75 – Вміст текстового файлу

Окрім шматочка прапора, цей файл також містив Нех-текст, з якого ми отримали QR-код.



Рисунок 3.76 – Отриманий QR-код

6) Далі перевіряючи зображення, знаходимо захований текстовий файл і ще одну частину прапора.

flag{Ev3ry_C0rn3r_of_0ur_Mus3um_H0lds_A_Un1qu3_St0ry_And_A_H1dd3n_S3cr3t_Fr0m_

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				125
Змн.	Арк.	№ докум.	Підпис	Дата		

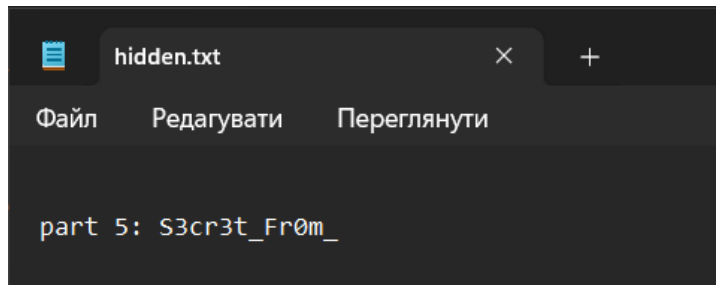


Рисунок 3.77 – Вміст текстового файлу

7) Відскануємо QR-код і отримаємо шосту частину прапора.

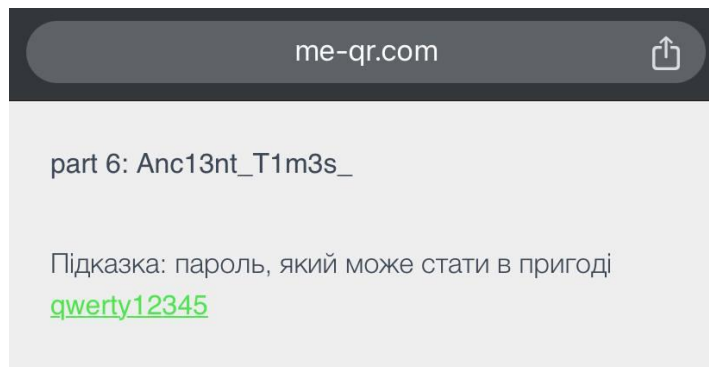


Рисунок 3.78 – Результат сканування QR-коду

8) За допомогою програми Openstego та отриманого паролю видобудемо прихований файл.

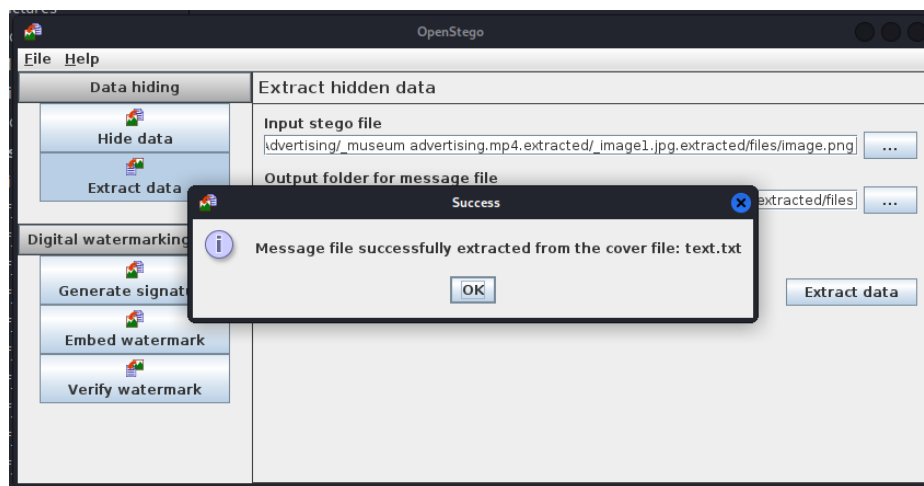


Рисунок 3.79 – Результат отримання файлу

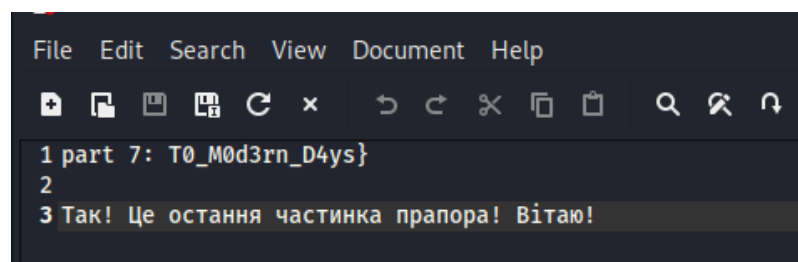


Рисунок 3.80 – Вміст текстового файлу

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				126
Змн.	Арк.	№ докум.	Підпис	Дата		

Вихідні дані:

flag{Ev3ry_C0rn3r_Of_Our_Mus3um_H0lds_A_Un1qu3_St0ry_And_A_H1dd3n_S3cr3t_Fr0m_Anc13nt_T1m3s_T0_M0d3rn_D4ys}

3.4.6 «Melodies of Life»

Мета реалізації:

Завдання зосереджене на використанні різних методів приховування інформації у аудіофайлах. Саме тому учасникам потрібно буде дослідити аудіофайл, щоб знайти всі заховані частини прапора.

Використані техніки: аудіо стеганографія, SSTV, спектрограма

Мови програмування/інструменти: DeepSound, QSSTV, Audacity, Paint, Coagula, Sonic Visualiser, <https://audiotrimmer.com/>

Вхідні дані:

- **Назва завдання:** «Melodies of Life»
- **Опис до завдання:**

У нашому музеї представлена унікальна виставка, присвячена музичному мистецтву. Вона містить колекцію аудіоекспонатів, кожен з яких розповідає свою історію. Найзагадковішим експонатом є «Мелодії Життя». Нещодавно місцевий культуролог виявив, що в цьому аудіофайлі заховані фрагменти прапора, який є ключем до розуміння важливої історичної події, пов'язаної з появою та розвитком музичного мистецтва.

Ваше завдання - використати всі доступні інструменти для аналізу та обробки аудіофайлів, щоб знайти приховані фрагменти і зібрати їх разом. Зрештою, ви розгадаєте загадку та відкриєте глибші сенси, що ховаються в музиці.

- **Складність:** складний рівень
- **Формат прапора:** після виконання завдання ви отримаєте прапор формату flag{Слово1_Слово2}
- **Файли для розв'язання завдання:**

Кроки реалізації:

- 1) Обираємо відповідний прапор та розділяємо його на три частини.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				127
Змн.	Арк.	№ докум.	Підпис	Дата		

Перша частина буде захована у спектрограмі аудіофайлу. Для цього варто в Paint створити картинку з прапором. Далі нам потрібно скористатися програмним забезпеченням Coagula, яке створює звук із зображення.

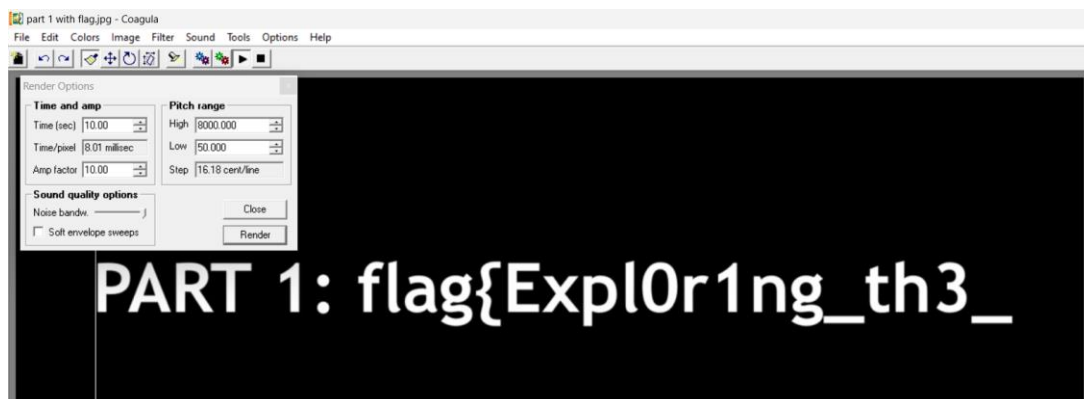


Рисунок 3.81 – Процес створення звуку

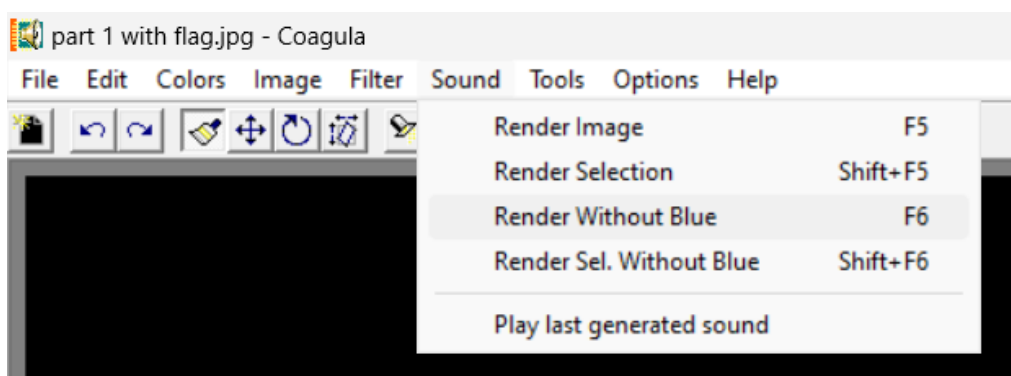


Рисунок 3.82 – Процес створення звуку

За допомогою програми Sonic Visualiser та шару Spectrogram (Спектрограма) переглянемо результат створення аудіофайлу.

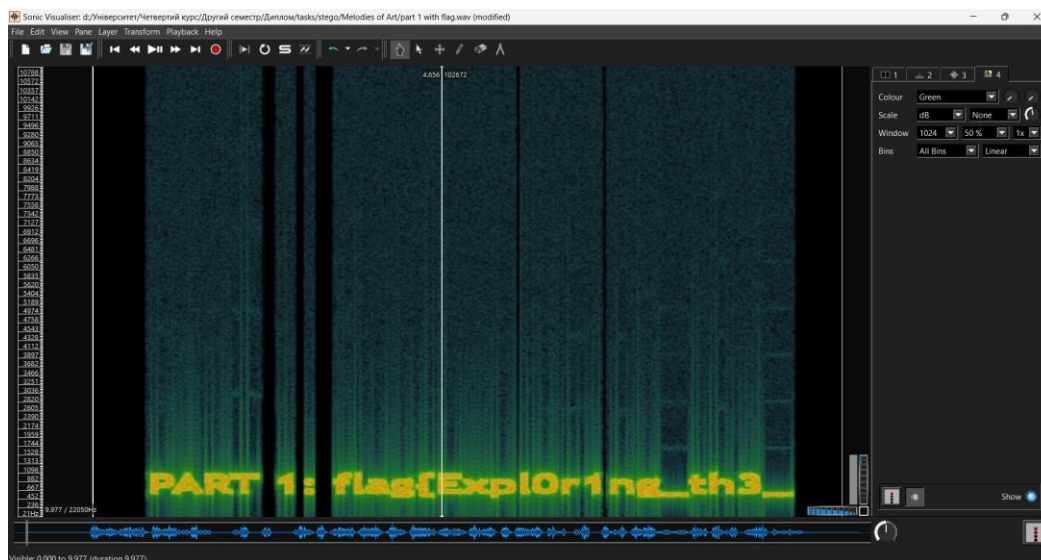


Рисунок 3.83 – Результат створення аудіофайлу

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				128
Змн.	Арк.	№ докум.	Підпис	Дата		

2) Для приховування іншої частини використаємо програму QSSTV. Передамо наше зображення з другою частиною прапору через радіохвилі і збережемо отриманий аудіофайл.

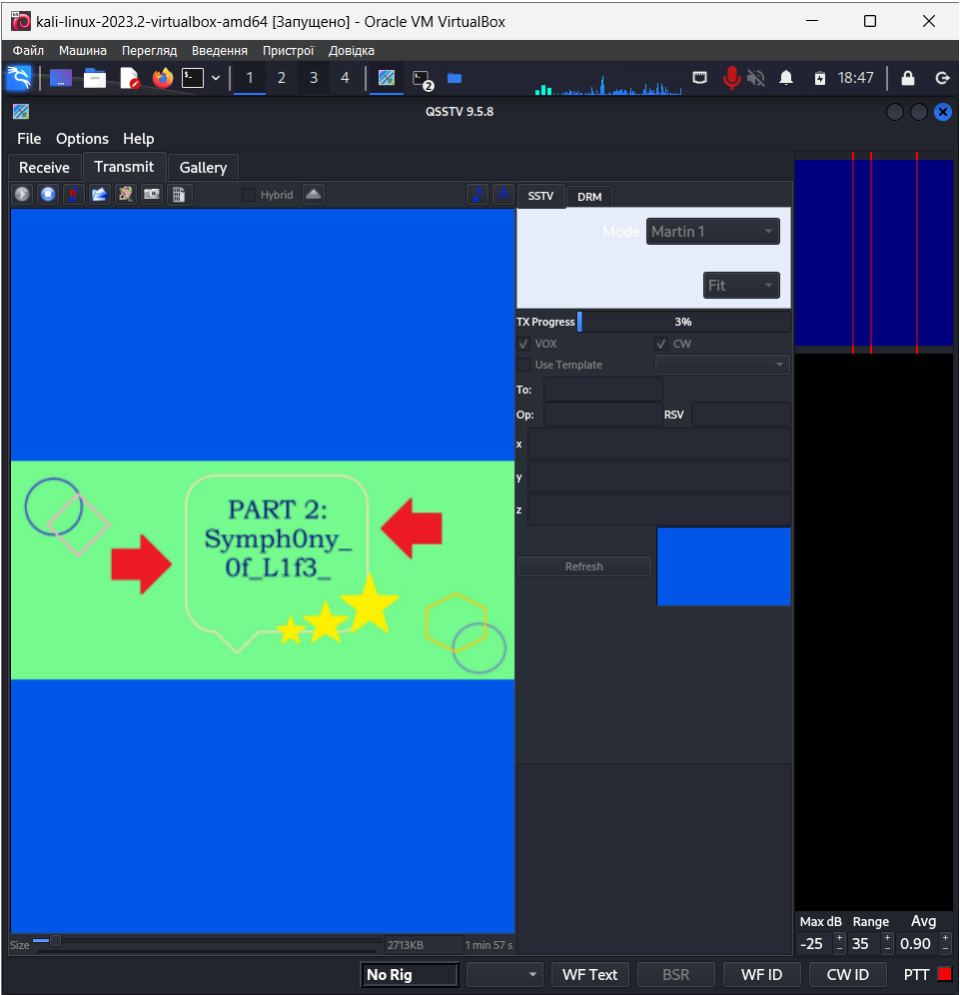


Рисунок 3.84 – Процес передачі зображення

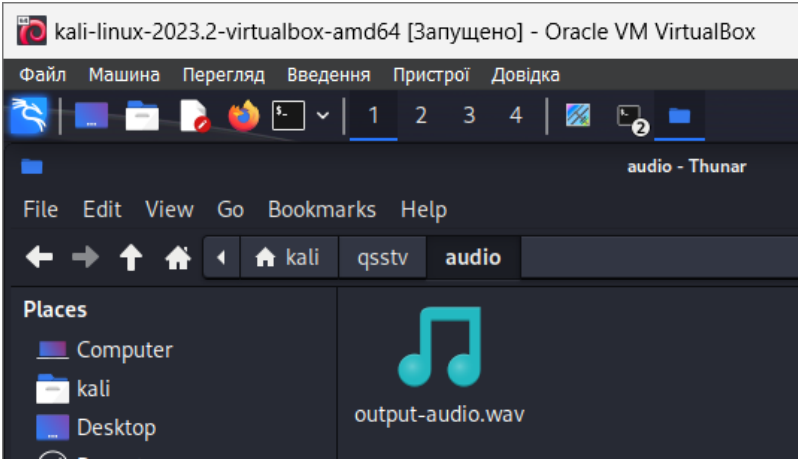


Рисунок 3.85 – Отриманий аудіофайл

Спробуємо декодувати отриманий аудіофайл назад у зображення.

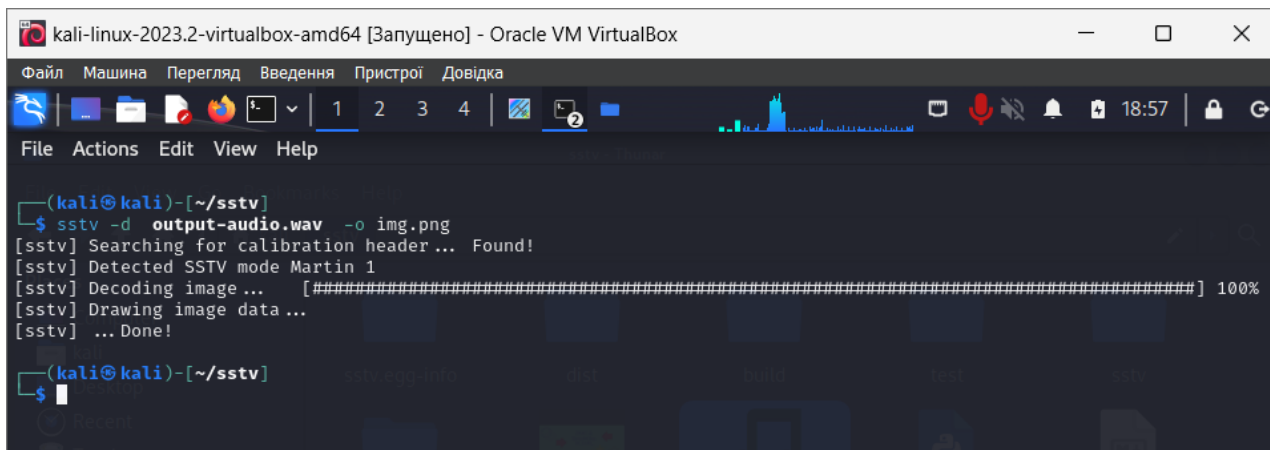


Рисунок 3.86 – Декодування аудіофайлу

- 3) Для створення загального аудіофайлу використаємо будь-яку мелодію і наріжемо кілька фрагментів по 30 секунд.

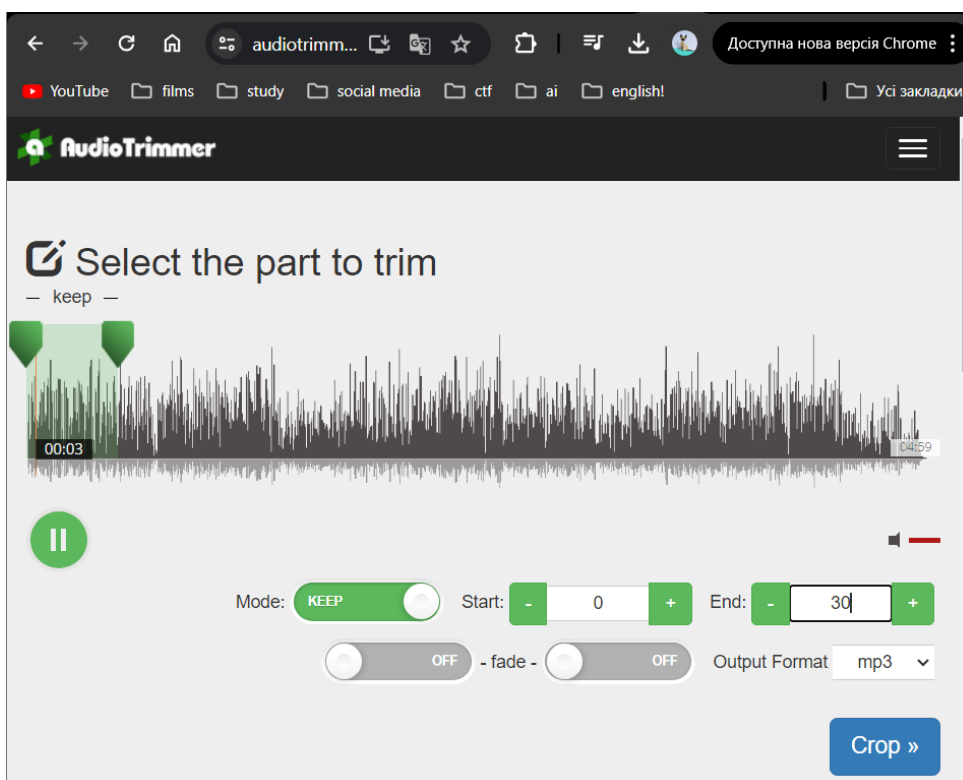


Рисунок 3.87 – Процес обрізання звуку

- 4) Побудуємо в програмі Audacity наш аудіофайл з частин, отриманих в попередніх кроках.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		130

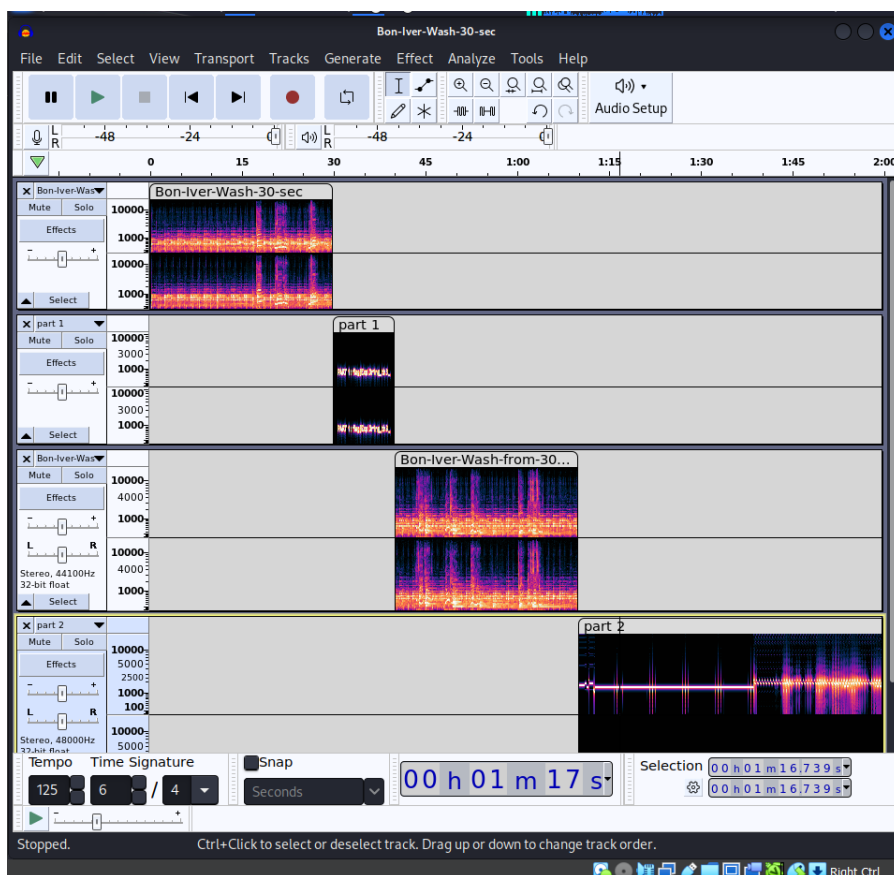


Рисунок 3.88 – Результат створення фінального аудіофайлу

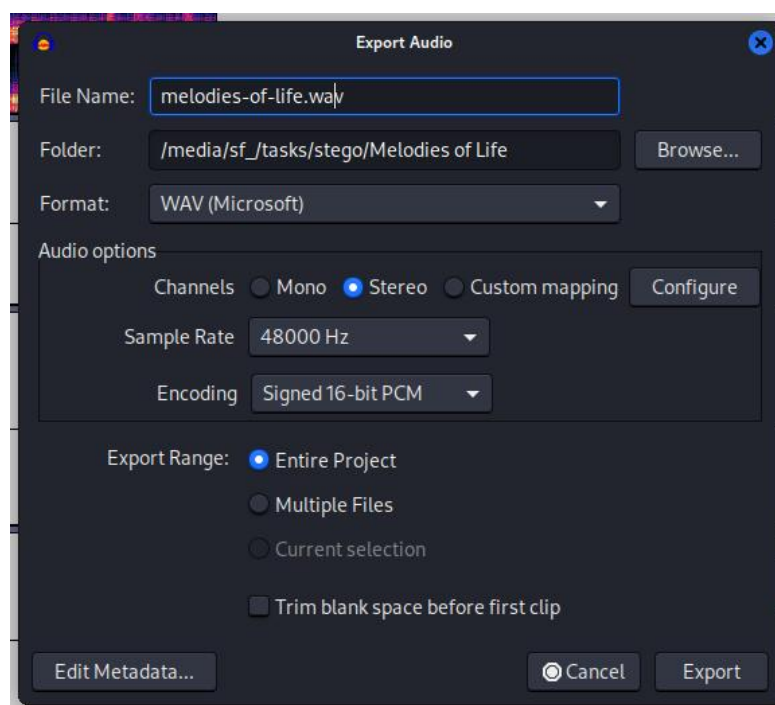


Рисунок 3.89 – Процес збереження аудіофайлу

5) В текстовому файлі збережемо третю частину прапора.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				131
Змн.	Арк.	№ докум.	Підпис	Дата		

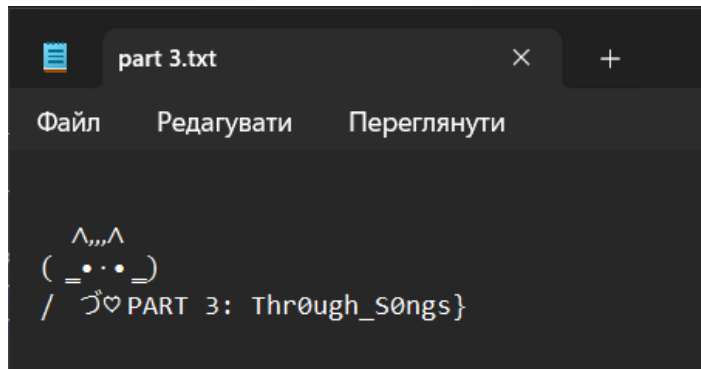


Рисунок 3.90 – Створення файлу з третьою частиною прапора

- 6) Далі за допомогою програми DeepSound приховуємо та зашифруємо створений текстовий файл в аудіо.

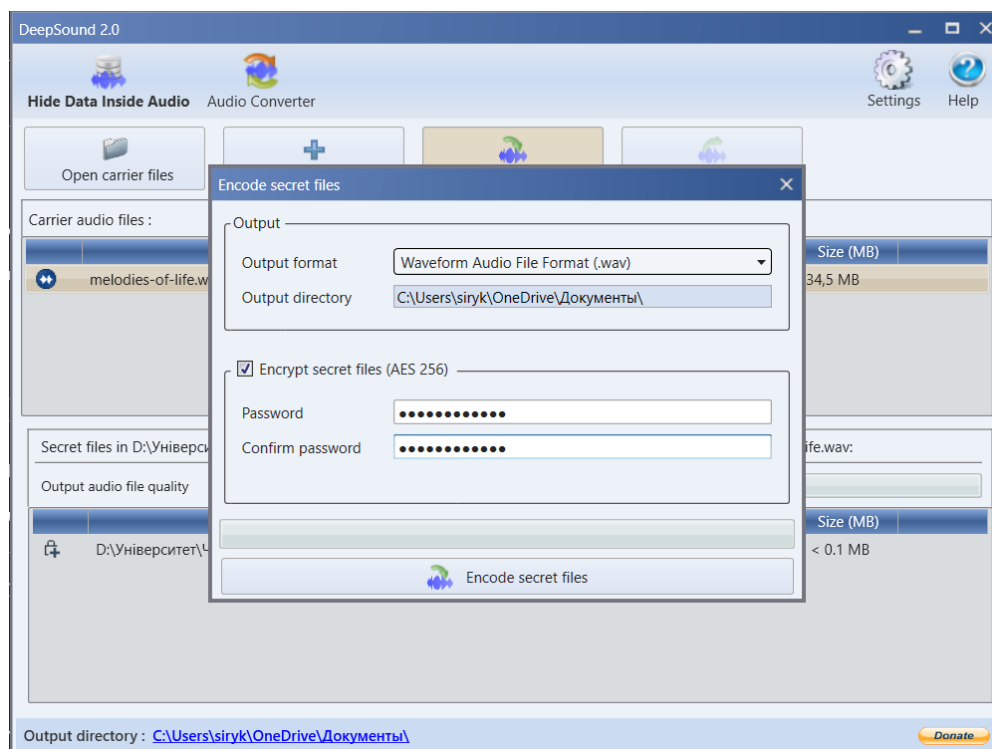


Рисунок 3.91 – Приховування текстового файлу

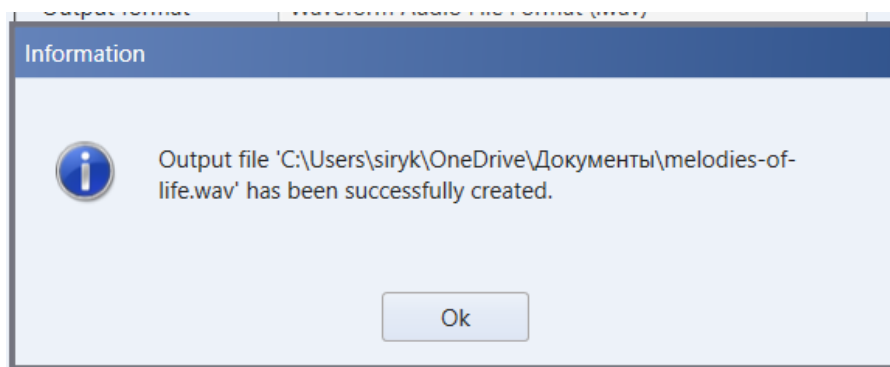


Рисунок 3.92 – Результат приховання файлу

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				132
Змн.	Арк.	№ докум.	Підпис	Дата		

Інструменти для рішення завдання: John The Ripper, DeepSound, SSTV, Sonic Visualiser

Тестування та перевірка:

- 1) Першим кроком, після прослуховування аудіофайлу, має бути перегляд його спектрограми. Тут ми можемо помітити першу частину прапора.
- 2) Далі декодування аудіофайлу за допомогою SSTV Decoder.
- 3) Останній етап – це отримати з аудіо текстовий файл із використанням програми DeepSound.

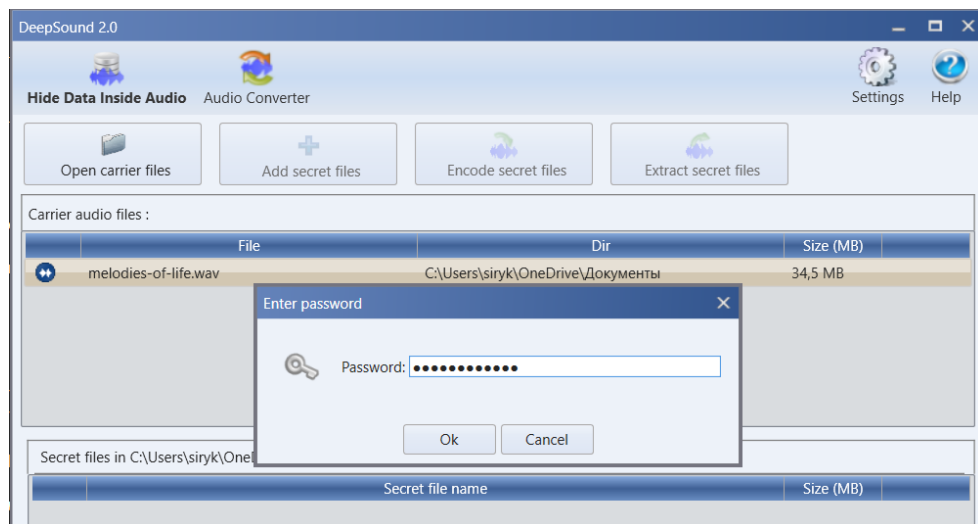


Рисунок 3.93 – Процес отримання текстового файлу

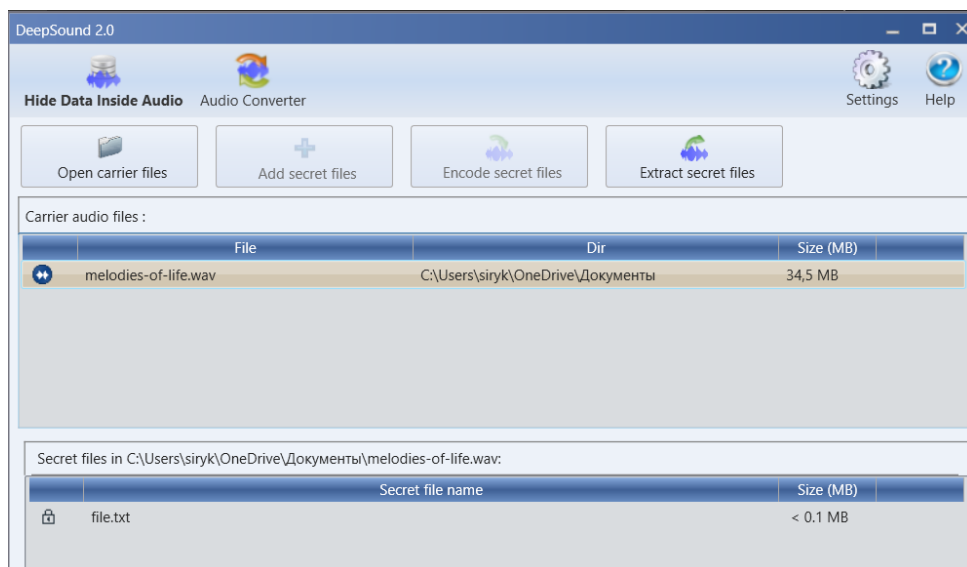


Рисунок 3.94 – Результат отримання текстового файлу

Ми з легкістю отримали файл, адже знаємо пароль. Однак, учасникам він невідомий тому, їм варто використати інструмент для брутфорсу паролю.

```

kali@kali:~/media/sf/tasks/stego/Melodies of Life$ python3 deepsound2john.py 'melodies-of-life.wav' > flag.txt
kali@kali:~/media/sf/tasks/stego/Melodies of Life$ cat flag.txt
melodies-of-life.wav:$dynamic_1529$4d021ae5c3852bccd8d3e84a6c4720b23e1eb134
kali@kali:~/media/sf/tasks/stego/Melodies of Life$ john --wordlist=/usr/share/wordlists/rockyou.txt --format=dynamic_1529 --fork=3 flag.txt
Using default input encoding: UTF-8
Loaded 1 password hash (dynamic_1529 [sha1($p null_padded_to_len_32) (DeepSound) 128/128 SSE2 4x1])
No password hashes left to crack (see FAQ)
kali@kali:~/media/sf/tasks/stego/Melodies of Life$ john flag.txt --show
melodies-of-life.wav:feelthemusic
1 password hash cracked, 0 left
kali@kali:~/media/sf/tasks/stego/Melodies of Life$

```

Рисунок 3.95 – Процес брутфорсу паролю

Вихідні дані: flag{Expl0r1ng_th3_Symph0ny_Of_L1f3_Thr0ugh_S0ngs}

Висновки до третього розділу

У третьому розділі було детально розглянуто та реалізовано ряд криптографічних та стеганографічних завдань для CTF-змагань, що охоплюють різні рівні складності: від легких до складних. Практична частина включала розробку завдань з використанням популярних криптографічних алгоритмів, таких як шифр Цезаря, Гілла, Віженера, та створення завдань на основі сучасних методів стеганографії.

Зокрема, розроблені завдання криптографічного характеру надають учасникам можливість ознайомитись із класичними методами шифрування, такими як «Julius Caesar», а також зануритись у більш складні концепції, представлені у завданнях «D1v3 Into Th3 D3pths!» та «Modern Art». Кожен виклик сприяє розвитку навичок у криптоаналізі та підвищенню рівня обізнаності щодо захисту інформації.

Серед стеганографічних завдань було розроблено цікаві та пізнавальні задачі, що охоплюють як прості методи приховування інформації, такі як у завданні «The Winged Victory of Samothrace», так і більш витончені техніки, представлені у викликах «Mona Lisa» та «Museum Advertising». Це дозволяє учасникам відчути на практиці різні підходи до приховування інформації та оцінити їх ефективність.

Отже, у третьому розділі детально розглянуто практичні аспекти реалізації криптографічних та стеганографічних завдань. Створені завдання сприяють розвитку аналітичного мислення, навичок роботи з сучасними інструментами та методами, а також підвищують загальний рівень знань у сфері інформаційної безпеки.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		135

ВИСНОВКИ

У результаті виконання випускної кваліфікаційної роботи було спроектовано, створено та протестовано комплексний навчальний набір завдань, спрямований на розвиток навичок у галузі Cryptography та Steganography для студентських CTF-змагань. Дослідження охоплювало теоретичний аналіз методів і засобів криптографії та стеганографії, розробку завдань різної складності, а також вибір і застосування відповідних інструментів для їх реалізації.

Перший розділ роботи забезпечив детальний огляд криптографічних та стеганографічних методів, які використовуються в CTF-змаганнях. Розглянуто основні поняття криптографії, класичні та сучасні алгоритми шифрування, а також методи приховування інформації у текстах, зображеннях, аудіо та відео. Ці теоретичні знання заклали фундамент для подальшої розробки практичних завдань.

Другий розділ був присвячений процесу проектування завдань, аналізу та вибору інструментів для їх розробки та вирішення. Було вивчено широкий спектр криптографічних і стеганографічних інструментів, включаючи онлайн-ресурси, бібліотеки Python та спеціалізовані програми. Розроблені рекомендації щодо використання цих інструментів дозволяють створювати завдання різної складності та забезпечують ефективність їх виконання учасниками змагань.

У третьому розділі здійснено практичну реалізацію криптографічних та стеганографічних завдань. Було створено низку викликів різної складності, від легких до складних. Криптографічні завдання включали класичні методи шифрування, такі як шифр Цезаря та Віженера, а також складніші алгоритми. Стеганографічні завдання охоплювали різні методи приховування інформації в медіафайлах, що дозволяє учасникам змагань відчувати практичні аспекти цих технологій.

Отже, випускна кваліфікаційна робота досягла поставленої мети – створено та протестовано навчальний набір завдань для студентського змагального CTF-майданчика, який є ефективним засобом для підвищення рівня знань студентів у сфері інформаційної безпеки. Ці завдання сприяють розвитку аналітичного мислення, вдосконаленню навичок роботи з сучасними інструментами та методами, а також готують студентів до участі у змаганнях з кібербезпеки.

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		136

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Vanstone S. A., Menezes A. J., Oorschot P. C. v. Handbook of Applied Cryptography. Taylor & Francis Group, 2018. 810 p.
2. Smart N. P. Cryptography Made Simple. Cham : Springer International Publishing, 2016. URL: <https://doi.org/10.1007/978-3-319-21936-3> (дата звернення: 20.04.2024).
3. Mollin R. A. An Introduction to Cryptography. Chapman and Hall/CRC, 2006. URL: <https://doi.org/10.1201/9781420011241> (дата звернення: 26.04.2024).
4. Henk C. A. van Tilborg. Encyclopedia of Cryptography and Security. 2-ге вид. Boston, MA : Springer Science+Business Media, LLC, 2011.
5. Serious Cryptography: A Practical Introduction to Modern Encryption. No Starch Press, 2017.
6. Paar C., Pelzl J. Understanding Cryptography. Berlin, Heidelberg : Springer Berlin Heidelberg, 2010. URL: <https://doi.org/10.1007/978-3-642-04101-3> (дата звернення: 20.04.2024).
7. КОМП'ЮТЕРНА СТЕГАНОГРАФІЯ - навчальний посібник. Головна - Репозитарій Вінницького Національного Технічного Університету. URL: <https://ir.lib.vntu.edu.ua/bitstream/handle/123456789/34907/90812.pdf?sequence=2∓isAllowed=y> (дата звернення: 20.04.2024).
8. Agarwal M. Text Steganographic Approaches: A Comparison. International Journal of Network Security & Its Applications. 2013. Т. 5, № 1. С. 91–106. URL: <https://doi.org/10.5121/ijnsa.2013.5107> (дата звернення: 20.04.2024).
9. CTFtime.org / What is Capture The Flag?. CTFtime.org / All about CTF (Capture The Flag). URL: <https://ctftime.org/ctf-wtf/> (дата звернення: 21.04.2024).
10. Beginner's Guide – SANReN Cyber Security Challenge. SANReN Cyber Security Challenge. URL: https://www.csc.ac.za/?page_id=555 (дата звернення: 21.04.2024).
11. Корченко О. Г., Сіденко В. П., Дрейс Ю. О. Прикладна криптологія: системи шифрування : підручник. Житомир, 2014. 448 с. URL: https://er.nau.edu.ua/bitstream/NAU/32583/1/ПК_системи%20шифрування.pdf.
12. CTF Cryptography for Beginners :: CharCharBoggles. CTF Cryptography for Beginners. URL: https://charcharbinks.com/post/ctf_crypto_for_beginners/ (дата звернення: 22.04.2024).
13. Encryption vs Encoding | Infosec. Cybersecurity Training & Certifications | Infosec. URL: <https://www.infosecinstitute.com/resources/cryptography/encryption-vs-encoding/> (дата звернення: 25.04.2024).
14. Types of Encoding Techniques - Javatpoint. www.javatpoint.com. URL: <https://www.javatpoint.com/types-of-encoding-techniques> (дата звернення: 25.04.2024).
15. Singh P. Types of Steganography methods that are used for hiding confidential data. LinkedIn. URL: <https://www.linkedin.com/pulse/types-steganography-methods-used-hiding-confidential-data-singh/> (date of access: 25.04.2024).

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		137

16. Nashat D., Mamdouh L. An efficient steganographic technique for hiding data. Journal of the Egyptian Mathematical Society. 2019. Vol. 27, no. 1. URL: <https://doi.org/10.1186/s42787-019-0061-6> (date of access: 07.05.2024).
17. Audio Steganography In More Detail. Sven Ruppert. URL: <https://svenruppert.com/2024/04/17/audio-steganography-in-more-detail/> (дата звернення: 02.05.2024).
18. How to hide secret messages in music files? | Deep Sound. IICS. URL: <https://www.iicybersecurity.com/audio-steganography.html> (дата звернення: 02.05.2024).
19. Audio Steganography - CTF Wiki EN. Getting Started - CTF Wiki EN. URL: <https://ctf-wiki.mahaloz.re/misc/audio/introduction/> (дата звернення: 02.05.2024).
20. Palriwala S. Composing CTF Challenges. medium. URL: <https://medium.com/techloop/composing-ctf-challenge-b5828dba0feb> (дата звернення: 02.05.2024).
21. Singh A. P., Moudgil S., Rani S. An Acquaintance to Text-Steganography and its Methods. Journal of Physics: Conference Series. 2021. Т. 1950, № 1. С. 012005. URL: <https://doi.org/10.1088/1742-6596/1950/1/012005> (дата звернення: 03.05.2024).
22. Agarwal M. Text Steganographic Approaches: A Comparison. International Journal of Network Security & Its Applications. 2013. Т. 5, № 1. С. 91–106. URL: <https://doi.org/10.5121/ijnsa.2013.5107> (дата звернення: 03.05.2024).
23. Hide information in image files using WxHexEditor. medium. URL: <https://medium.com/purple-team/hide-information-in-image-files-using-wxhexeditor-744618d6a181> (дата звернення: 07.05.2024).
24. Bhayani A. Everything that you need to know about Image Steganography. LinkedIn. URL: <https://www.linkedin.com/pulse/everything-you-need-know-image-steganography-arpit-bhayani-qedhc/> (дата звернення: 08.05.2024).
25. GitHub - RsaCtfTool/RsaCtfTool: RSA attack tool (mainly for ctf) - retrieve private key from weak public key and/or uncipher data. GitHub. URL: <https://github.com/RsaCtfTool/RsaCtfTool> (дата звернення: 08.05.2024).
26. File Magic Numbers. Gist. URL: <https://gist.github.com/leommoore/f9e57ba2aa4bf197ebc5> (дата звернення: 09.05.2024).
27. GitHub - devploit/awesome-ctf-resources: A list of Capture The Flag (CTF) frameworks, libraries, resources and software for started/experienced CTF players. GitHub. URL: <https://github.com/devploit/awesome-ctf-resources?tab=readme-ov-file> (дата звернення: 09.05.2024).
28. NumPy documentation – NumPy v1.26 Manual. NumPy -. URL: <https://numpy.org/doc/stable/> (дата звернення: 15.05.2024).
29. Crypto. PyPI Package and Documentation Storage. URL: <https://pythonhosted.org/pycrypto/Crypto-module.html> (дата звернення: 15.05.2024).
30. Welcome to gmpy2's documentation! – gmpy2 2.2.0a2 documentation. URL: <https://gmpy2.readthedocs.io/en/latest/> (дата звернення: 15.05.2024).
31. Steganography Toolkit. GitHub. URL: <https://github.com/DominicBreuker/stego-toolkit> (дата звернення: 16.05.2024).

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		138

32. Steganography | 0xffd700. 0xffd700 - Tech diary. | 0xffd700. URL: <https://0xffd700.com/steganography> (дата звернення: 16.05.2024).
33. SSTV Tools. DX Atlas: Amateur Radio software. URL: <https://www.dxatlas.com/sstvtools/> (дата звернення: 16.05.2024).
34. Advances in Cryptology - EUROCRYPT 2007 / ред. М. Naor. Berlin, Heidelberg : Springer Berlin Heidelberg, 2007. URL: <https://doi.org/10.1007/978-3-540-72540-4> (дата звернення: 31.05.2024).
35. G R M., R B S. Video Steganography: A Survey of Techniques and Methodologies. SSRN Electronic Journal. 2021. URL: <https://doi.org/10.2139/ssrn.3851241> (дата звернення: 01.06.2024).
36. Weir J., Yan W. A Comprehensive Study of Visual Cryptography. Transactions on Data Hiding and Multimedia Security V. Berlin, Heidelberg, 2010. С. 70–105. URL: https://doi.org/10.1007/978-3-642-14298-7_5 (дата звернення: 03.05.2024).
37. GitHub - ON4QZ/QSSTV: Receive and transmit images over radio using analog SSTV or digital DRM. GitHub. URL: <https://github.com/ON4QZ/QSSTV> (дата звернення: 17.05.2024).
38. GitHub - colaclanth/sstv: SSTV Decoder. GitHub. URL: <https://github.com/colaclanth/sstv> (дата звернення: 17.05.2024).
39. fcrackzip | Kali Linux Tools. Kali Linux. URL: <https://www.kali.org/tools/fcrackzip/> (дата звернення: 01.06.2024).
40. Cracking a password-protected zip file with john - Tech Couch. Tech Couch. URL: <https://tech-couch.com/post/cracking-a-password-protected-zip-file-with-john> (дата звернення: 01.06.2024).
41. GitHub - Assa228/StegsnowBruteForcer: Stegsnow password Bruteforcer - Steganography brute-force utility to uncover hidden data inside files. GitHub. URL: <https://github.com/Assa228/StegsnowBruteForcer> (дата звернення: 20.05.2024).
42. GitHub - 0xHasanM/SnowCracker: Snowcracker used to crack Stegsnow passwords. GitHub. URL: <https://github.com/0xHasanM/SnowCracker> (дата звернення: 21.05.2024).
43. Unicode Steganography with Zero-Width Characters. GitHub Pages. URL: https://330k.github.io/misc_tools/unicode_steganography.html (дата звернення: 19.05.2024).

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				139
Змн.	Арк.	№ докум.	Підпис	Дата		

ДОДАТКИ

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				140
Змн.	Арк.	№ докум.	Підпис	Дата		

Лістинг до завдань з криптографії

Лістинг коду create_caesar.py для завдання «Julius Caesar»

```
# Функція шифрування тексту за допомогою шифру Цезаря з заданим ключем
def caesar_cipher(text, key):

    #Оголошуємо алфавіт, що містить цифри та літери англійського алфавіту верх-
    #нього та нижнього регістрів
    alphabet = "0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz"

    #Ініціалізуємо порожню змінну для збереження результату шифрування
    result = ""

    #Проходимося по кожному символу у вхідному тексті
    for char in text:
        #Перевіряємо, чи символ знаходиться в алфавіті
        if char in alphabet:
            #Знаходимо індекс символу в алфавіті й зсуваємо його на ключ
            shifted_index = (alphabet.index(char) + key) % len(alphabet)

            #Додаємо зсунутий символ до результату
            result += alphabet[shifted_index]
        else:
            #Якщо символ не знайдено в алфавіті, додаємо його без змін
            result += char

    #Повертаємо зашифрований текст.
    return result

#Вхідне повідомлення, яке буде зашифровано
message = "flag{1_C4m3_1_S4w_1_C0nqu3r3d}"

#Зашифровуємо вхідне повідомлення з використанням функції caesar_cipher і ключа 19
encrypted_message = caesar_cipher(message, 19)

#Виводимо зашифроване повідомлення
print("Зашифроване повідомлення: ", encrypted_message)
```

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		141

Лістинг коду create_hill.py для завдання «Lovely Hill Picture»

```
import numpy as np

#Функція для шифрування за допомогою шифру Гілла
def hill_cipher_encrypt(plaintext, key_matrix):
    #Перетворення тексту в числовий вектор (A=0, B=1, ..., Z=25)
    plaintext = plaintext.replace(" ", "").upper()
    plaintext = [ord(char) - 65 for char in plaintext]

    #Додавання символу 'X', якщо довжина вхідного тексту не кратна розміру ключо-
    вої матриці
    while len(plaintext) % key_matrix.shape[0] != 0:
        plaintext.append(23) #23 - код доповнювального символу ('X')

    #Поділ тексту на блоки та шифрування
    ciphertext = ""
    for i in range(0, len(plaintext), key_matrix.shape[0]):
        block = plaintext[i:i + key_matrix.shape[0]]
        block = np.array(block)
        encrypted_block = np.dot(key_matrix, block) % 26
        ciphertext += ''.join([chr(char + 65) for char in encrypted_block])

    return ciphertext

# Функція для збереження зашифрованого тексту у файл
def save_to_file(filename, data):
    with open(filename, "w") as file:
        file.write(data)

#Відкритий текст та ключ для шифрування
plaintext = "IREALLYLIKETHISCALIFORNIAHILL"
key = "LANDSCAPE"

#Побудова ключова матриця
key_size = int(np.sqrt(len(key)))
key_matrix = np.array([[ord(char) - 65 for char in key[i:i + key_size]] for i in
range(0, len(key), key_size)])

print("Ключова матриця: \n",key_matrix)

#Шифрування
ciphertext = hill_cipher_encrypt(plaintext, key_matrix)
print("Зашифрований прапор:", ciphertext)

# Збереження зашифрованого тексту у файл hill_text.txt
save_to_file("hill_text.txt", ciphertext)
```

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				142
Змн.	Арк.	№ докум.	Підпис	Дата		

Лістинг коду *create_vigenere.py* для завдання «*Vigenère Quest*»

```
def vigenere_cipher(plain_text, key):
    cipher_text = ""
    key_index = 0
    for char in plain_text:
        if char.isalpha():
            # Знаходимо номер символу у ключі
            key_char = key[key_index % len(key)]
            key_shift = ord(key_char.lower()) - ord('a')
            # Шифруємо символ
            if char.islower():
                cipher_char = chr((ord(char) - ord('a') + key_shift) % 26 +
ord('a'))
            else:
                cipher_char = chr((ord(char) - ord('A') + key_shift) % 26 +
ord('A'))
            cipher_text += cipher_char
            key_index += 1
        else:
            # Якщо символ не буква, просто додаємо його до зашифрованого тексту
            cipher_text += char
    return cipher_text

# Список слів для шифрування
words = ["explorer", "hidden", "route", "expedition", "secret", "discovery",
"legend", "ancient",
        "code", "mystery", "journey", "artifacts", "adventure", "manuscripts",
"compass",
        "chronicle", "coordinates", "treasure",
        "the treasure that you seek is hidden in the place you least expect"]

# Початковий ключ
key = "vigenere"
previous_word = ""

# Відкриття файлу для запису зашифрованих слів та ключа
with open("cipher.txt", "w") as file:
    # Записуємо ключ у файл
    file.write("Key -> vigenere\n\n")
    # Шифруємо кожне слово
    for i, word in enumerate(words, 1):
        if previous_word:
            key = previous_word
        encrypted_word = vigenere_cipher(word, key)
        print("Ciphertext", i, "->", encrypted_word)
        # Записуємо зашифроване слово у файл
        file.write(f"Ciphertext {i} -> {encrypted_word}\n")
        previous_word = word

print("\nEncrypted words saved to cipher.txt")
```

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				143
Змн.	Арк.	№ докум.	Підпис	Дата		

Лістинг коду solve_vigenere.py для завдання «Vigenère Quest»

```
def vigenere_cipher(plain_text, key):
    cipher_text = ""
    key_index = 0
    for char in plain_text:
        if char.isalpha():
            # Знаходимо номер символу у ключі
            key_char = key[key_index % len(key)]
            key_shift = ord(key_char.lower()) - ord('a')
            # Шифруємо символ
            if char.islower():
                cipher_char = chr((ord(char) - ord('a') + key_shift) % 26 +
ord('a'))
            else:
                cipher_char = chr((ord(char) - ord('A') + key_shift) % 26 +
ord('A'))
            cipher_text += cipher_char
            key_index += 1
        else:
            # Якщо символ не буква, просто додаємо його до зашифрованого тексту
            cipher_text += char
    return cipher_text

def vigenere_decipher(cipher_text, key):
    plain_text = ""
    key_index = 0
    for char in cipher_text:
        if char.isalpha():
            # Знаходимо номер символу у ключі
            key_char = key[key_index % len(key)]
            key_shift = ord(key_char.lower()) - ord('a')
            # Розшифровуємо символ
            if char.islower():
                plain_char = chr((ord(char) - ord('a') - key_shift + 26) % 26 +
ord('a'))
            else:
                plain_char = chr((ord(char) - ord('A') - key_shift + 26) % 26 +
ord('A'))
            plain_text += plain_char
            key_index += 1
        else:
            # Якщо символ не буква, просто додаємо його до розшифрованого тексту
            plain_text += char
    return plain_text

# Використання функції розшифрування
# Відкриття файлу для читання зашифрованих слів
with open("cipher.txt", "r") as file:
    lines = file.readlines()

# Читання ключа
key = lines[0].split("->")[1].strip()

# Розшифровуємо кожне зашифроване слово
decrypted_words = []
for line in lines[2:]:
    encrypted_word = line.split("->")[1].strip()
    decrypted_word = vigenere_decipher(encrypted_word, key)
    decrypted_words.append(decrypted_word)
    key = decrypted_word # Оновлюємо ключ для наступного слова

# Виводимо розшифровані слова
```

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				144
Змн.	Арк.	№ докум.	Підпис	Дата		

```

for i, word in enumerate(decrypted_words, 1):
    print(f"Plaintext {i} -> {word}")

# Виводимо останнє розшифроване повідомлення у форматі flag{слово1_слово2}
if decrypted_words:
    # Розбиваємо останнє розшифроване повідомлення на слова і об'єднуємо їх символом підкреслення
    last_decrypted_message = decrypted_words[-1].replace(" ", "_")
    flag = f"flag{{{last_decrypted_message}}}"
    print(f"\nFlag: {flag}")

```

Лістинг коду create_stars.py для завдання «The Starry Bytes»

```

from PIL import Image
from itertools import cycle

def xor_bytes(a, b):
    return bytes(i ^ j for i, j in zip(a, cycle(b)))

# Відкриваємо зображення для зашифрування та отримуємо дані
with open("The Starry Night (original) with flag.png", "rb") as file:
    data = file.read()
    key = data[:8]

# Виконуємо операцію XOR для зашифрування даних
encrypted_data = xor_bytes(data, key)

# Записуємо зашифровані дані у файл 'encrypted_stars.txt'
with open('encrypted_stars.txt', 'wb') as output_file:
    output_file.write(encrypted_data)

```

Лістинг коду solve_stars.py для завдання «The Starry Bytes»

```

from PIL import Image
from itertools import cycle

def xor(a, b):
    return [i^j for i, j in zip(a, cycle(b))]

# Зчитуємо дані, зашифровані у файлі 'encrypted_stars.txt'
enc_data = open('encrypted_stars.txt', 'rb').read()

# Задаємо ключ key = [89, 50, 4E, 47, 0d, 0a, 1a, 0a]
key = [0x89, 0x50, 0x4E, 0x47, 0x0d, 0x0a, 0x1a, 0x0a]

# Виконуємо операцію XOR для розшифрування даних
decrypted_data = bytearray(xor(enc_data, key))

# Записуємо розшифровані дані у файл 'decrypted_stars_picture.png'
open('decrypted_stars_picture.png', 'wb').write(decrypted_data)

```

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		145

Лістинг коду create_rsa.py для завдання «Rhythmic Symphony of Art»

```
from Crypto.Util.number import *

flag = b"flag{S0rry_F0r_Th1s_L4y3rs!_But_RSA_1s_My_H1dd3n_P4ss10n}"
# print(flag)

# Повідомлення, яке ми шифруємо
m = bytes_to_long(flag)
# print(m)

# Публічний експонент
e = 65537

# Генеруємо два великих простих числа
# p = getPrime(256)
p = 70193440209237683281120079211804319436238252095515166406003793897457294753803
print("p = ",p)
# q = getPrime(256)
q = 58831512050670440996995211777460393574318331023756409649104495052313173035561
print("q = ",q)

# Обчислюємо модуль n
n = p * q
print("n = ",n)

# Шифруємо повідомлення
cipher = pow(m, e, n)
print ("Зашифрований прапор: ",cipher)

# Записуємо числа n, e та cipher у файл
with open("rhythmic_symphony_of_art.txt", "w") as file:
    file.write(f"n = {n}\n")
    file.write(f"e = {e}\n")
    file.write(f"ciphertext = {cipher}\n")
```

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		146

Лістинг коду solve_rsa.py для завдання «Rhythmic Symphony of Art»

```
from Crypto.Util.number import inverse, long_to_bytes
import requests

# Задані значення
n =
4129586223547781845303704576822642310482658616794998412961819452175847026583316225
959777485813547234949494884110317929980677010038374279840167605658988483
e = 65537
ciphertext =
6766790564655781337962141254071388246751028690546681928961549527521366769588320157
86135426753762531272093268461713104077290960793177825390402028406593333

# Функція для отримання простих множників з factordb
def factor_n(n):
    url = f"http://factordb.com/api?query={n}"
    response = requests.get(url)
    data = response.json()
    factors = data.get("factors")
    if factors and len(factors) >= 2:
        return int(factors[0][0]), int(factors[1][0]) # Перетворюємо рядки на
цілі числа
    else:
        return None, None

# Розрахунок приватного ключа
def calculate_private_key(p, q, e):
    phi = (p - 1) * (q - 1)
    d = inverse(e, phi)
    return d

# Розшифрування шифротексту
def decrypt(ciphertext, n, d):
    plaintext = pow(ciphertext, d, n)
    return plaintext

# Знаходимо прості множники
p, q = factor_n(n)

# Розраховуємо приватний ключ
d = calculate_private_key(p, q, e)
print("Приватний ключ: ", d)

# Розшифровуємо шифротекст
flag = long_to_bytes(decrypt(ciphertext, n, d))

print("Розшифрований прапор: ", flag)
```

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				147
Змн.	Арк.	№ докум.	Підпис	Дата		

Лістинг коду create_visual_crypto.py для завдання «Modern Art»

```
from PIL import Image
import numpy as np

def create_shares(image_path, output1_path, output2_path):
    # Завантаження оригінального зображення
    image = Image.open(image_path)
    image = image.convert("1") # Конвертуємо в чорно-біле зображення
    pixels = np.array(image)

    # Створення пустих шарів
    share1 = np.zeros(pixels.shape, dtype=np.uint8)
    share2 = np.zeros(pixels.shape, dtype=np.uint8)

    # Генерація шарів
    for i in range(pixels.shape[0]):
        for j in range(pixels.shape[1]):
            pixel = pixels[i, j]
            if pixel == 0: # чорний піксель
                r = np.random.randint(2)
                share1[i, j] = r
                share2[i, j] = r
            else: # білий піксель
                r = np.random.randint(2)
                share1[i, j] = r
                share2[i, j] = 1 - r

    # Збереження шарів як зображення
    share1_img = Image.fromarray(share1 * 255) # Масштабування до діапазону [0, 255]
    share2_img = Image.fromarray(share2 * 255)

    share1_img.save(output1_path)
    share2_img.save(output2_path)

    print("Шари зображення створено та збережено!")

# Використання функції
create_shares('modern_art.png', 'painting1.png', 'painting2.png')
```

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				148
Змн.	Арк.	№ докум.	Підпис	Дата		

Лістинг коду solve_visual_crypto.py для завдання «Modern Art»

```
from PIL import Image
import numpy as np

def combine_shares(share1_path, share2_path, output_path):
    # Завантаження обох шарів
    share1 = Image.open(share1_path)
    share2 = Image.open(share2_path)

    # Конвертація шарів в чорно-білі зображення
    share1 = share1.convert("1")
    share2 = share2.convert("1")

    # Перетворення зображень у numpy масиви
    share1_pixels = np.array(share1)
    share2_pixels = np.array(share2)

    # Накладання шарів один на одного
    combined_pixels = np.logical_or(share1_pixels, share2_pixels)

    # Створення зображення з об'єднаних пікселів
    combined_image = Image.fromarray(np.uint8(combined_pixels) * 255)

    # Збереження об'єднаного зображення
    combined_image.save(output_path)

    print("Шари зображення об'єднано та збережено!")

# Використання функції
combine_shares('painting1.png', 'painting2.png', 'combined_image1.png')
```

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				149
Змн.	Арк.	№ докум.	Підпис	Дата		

Лістинг коду create_aes.py для завдання «Art Enthusiasts' Seminar»

```
from base64 import b64encode, b64decode
from hashlib import pbkdf2_hmac
from Crypto.Cipher import AES
from Crypto.Random import get_random_bytes
import os

# Вхідні дані
# Artist
password = 'artist10'.encode('utf-8') # Пароль
salt = get_random_bytes(16)           # Генерація випадкової солі
iv = get_random_bytes(16)             # Генерація випадкового IV
message = 'flag{AES_3ncRyp710n_15_4N_4r7_f0r_M45T3r_C0d3r5}'.encode('utf-8') #
Повідомлення для шифрування

# Генерація ключа за допомогою PBKDF2
key = pbkdf2_hmac('sha256', password, salt, 1000, dklen=32)

# Додавання паддингу (PKCS7)
pad_len = 16 - len(message) % 16
message += bytes([pad_len] * pad_len)

# Створення AES шифру та шифрування повідомлення
cipher = AES.new(key, AES.MODE_CBC, iv)
encrypted_message = cipher.encrypt(message)

# Перетворення на base64 для зручного представлення
encrypted_message_b64 = b64encode(encrypted_message).decode('utf-8')
salt_b64 = b64encode(salt).decode('utf-8')
iv_b64 = b64encode(iv).decode('utf-8')

print("Зашифроване повідомлення (base64):", encrypted_message_b64)
print("Сіль (base64):", salt_b64)
print("IV (base64):", iv_b64)

# Збереження даних у файл output.txt
with open('output.txt', 'w') as f:
    f.write("Ciphertext: " + encrypted_message_b64 + "\n")
    f.write("Salt: " + salt_b64 + "\n")
    f.write("IV: " + iv_b64 + "\n")
```

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		150

Лістинг коду sovlе_aes_brute_force_password.py для завдання «Art Enthusiasts' Seminar»

```
from base64 import b64decode
from hashlib import pbkdf2_hmac
from Crypto.Cipher import AES

# Читання зашифрованого тексту, солі та IV з файлу
with open('output.txt', 'r') as f:
    lines = f.readlines()
    encrypted_message_b64 = lines[0].split(": ")[1].strip()
    salt_b64 = lines[1].split(": ")[1].strip()
    iv_b64 = lines[2].split(": ")[1].strip()

# Декодування base64
encrypted_message = b64decode(encrypted_message_b64)
salt = b64decode(salt_b64)
iv = b64decode(iv_b64)

password_file = 'rockyou.txt' # Файл зі списком можливих паролів

# Відкриття файлу з паролями та перебір можливих паролів
with open(password_file, 'r') as f:
    for line in f:
        password = line.strip().encode('utf-8')
        try:
            # Генерація ключа за допомогою PBKDF2
            key = pbkdf2_hmac('sha256', password, salt, 1000, dklen=32)

            # Створення AES шифру та розшифрування повідомлення
            cipher = AES.new(key, AES.MODE_CBC, iv)
            decrypted_message = cipher.decrypt(encrypted_message)

            # Видалення падінгу (PKCS7)
            pad_len = decrypted_message[-1]
            decrypted_message = decrypted_message[:-pad_len]

            # Спроба декодувати повідомлення
            decrypted_message_str = decrypted_message.decode('utf-8')

            if 'flag' in decrypted_message_str:
                print("Знайдено пароль:", password.decode('utf-8'))
                print("Розшифроване повідомлення:", decrypted_message_str)
                break
        except (ValueError, KeyError):
            # Помилка при розшифруванні або видаленні падінгу, продовжуємо з наступним паролем
            continue
```

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		151

Лістинг до завдань з стеганографія

Лістинг коду add_metadata_comment_to_image.py для завдання «The Winged Victory of Samothrace»

```

from PIL import Image
from PIL.ExifTags import TAGS, GPSTAGS

def add_comment_to_image(image_path, new_image_name, comment):
    image = Image.open(image_path)

    # Перевірка чи формат зображення підтримує метадані
    if not image.format or not image.format.lower() in ["jpeg", "jpg"]:
        print("Цей формат зображення не підтримує метадані.")
        return

    # Отримати наявні метадані
    metadata = dict(image.info)

    # Переведення коментаря у формат ASCII
    ascii_comment = " ".join(str(ord(char)) for char in comment)
    print("Прапор у форматі ASCII: ", ascii_comment)

    # Додавання коментаря до метаданих
    metadata["comment"] = ascii_comment

    # Збереження зображення з оновленими метаданими та новою назвою
    print("Збережено метадані у", new_image_name)
    new_image_path = f"{new_image_name}.jpg"
    image.save(new_image_path, **metadata)

image_path = "The Winged Victory of Samothrace (original).jpg"
new_image_name = "The_Winged_Victory_of_Samothrace"
comment = "flag{W1ng3d_V1ct0ry_S3cr3t}"

add_comment_to_image(image_path, new_image_name, comment)

```

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		152

Лістинг коду `copy_images_change_time.py` для завдання «Mona Lisa»

```
import os
import shutil
import datetime

def change_creation_date(file_path, new_date):
    # Змінюємо дату створення файлу
    os.utime(file_path, (new_date.timestamp(), new_date.timestamp()))

def create_image_copies(original_images, num_copies):
    # Шлях, де будуть збережені копії картинок
    destination_folder = "D:\\Університет\\Четвертий курс\\Другий семестр\\Дип-
лом\\tasks\\stego\\Mona Lisa\\task_folder\\Art Collection of Mona Lisa"

    # Створюємо папку для копій, якщо вона ще не існує
    if not os.path.exists(destination_folder):
        os.makedirs(destination_folder)

    # Початкова дата створення/зміни для копій
    start_date = datetime.datetime(2010, 11, 19, 9, 30, 2)

    for i in range(num_copies):
        idx = i % len(original_images) # Визначаємо індекс поточної картинки
        image_path = original_images[idx]

        copy_path = os.path.join(destination_folder, f"Mona_Lisa_{i+1}.jpg")
        shutil.copyfile(image_path, copy_path)
        print(f"Створено копію Mona_Lisa_{i+1}: {copy_path}")

        # Змінюємо дату створення копії
        change_creation_date(copy_path, start_date)

if __name__ == "__main__":
    # Шляхи до оригінальних картинок, які потрібно скопіювати
    original_image_paths = ["Mona Lisa0.jpg", "Mona Lisa.jpg"]

    # Кількість копій, які треба створити для обох картинок
    num_copies = 2000

    # Створюємо копії картинок і змінюємо їхню дату створення
    create_image_copies(original_image_paths, num_copies)
```

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				153
Змн.	Арк.	№ докум.	Підпис	Дата		

Лістинг коду create_qr-code.py для завдання «Trojan Horse»

```
import os
from PIL import Image

# Завантаження QR-коду
qr_code = Image.open("trojan_horse_qr-code.png")

# Розміри QR-коду
width, height = qr_code.size

# Розмір шаблону для вирізання
piece_size = 25
rows = height // piece_size
cols = width // piece_size

# Створення папки для зберігання шматочків
output_folder = "mystery_pieces"
os.makedirs(output_folder, exist_ok=True)

# Вирізання та збереження шматочків
count = 0
for row in range(rows):
    for col in range(cols):
        left = col * piece_size
        top = row * piece_size
        right = left + piece_size
        bottom = top + piece_size
        piece = qr_code.crop((left, top, right, bottom))
        piece.save(os.path.join(output_folder, f"mystery_piece_{count}.png"))
        count += 1
```

Лістинг коду solve_qr-code.py для завдання «Trojan Horse»

```
from PIL import Image

# Створення нового зображення розміром 500x500
combined_image = Image.new('RGB', (500, 500))

# Перевірка, чи є всі 400 шматочків картинки
missing_pieces = []
for i in range(400):
    try:
        piece = Image.open(f'mystery_piece_{i}.png')
        if piece.size != (25, 25):
            raise ValueError(f"Розмір шматочка {i} не відповідає 25x25.")
        combined_image.paste(piece, ((i % 20) * 25, (i // 20) * 25))
    except FileNotFoundError:
        missing_pieces.append(i)

if missing_pieces:
    print(f"Не вдалося знайти шматочки з номерами: {missing_pieces}")
else:
    combined_image.save('combined_image.png')
    print("Об'єднане зображення збережено як 'combined_image.png'")
```

		Сірик А.В.			ККІКБ.КР.Б – 125 – 18 – ПЗ	Арк.
		Воротніков В.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		154