Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)

Навчально-науковий інститут деревообробних та
комп'ютерних технологій і дизайну
(повне найменування інституту, назва факультету (відділення))

Кафедра інформаційних технологій
(повна назва кафедри (предметної, циклової комісії))

Пояснювальна записка

до дипломної роботи
другий (магістерський)
(рівень вищої освіти)

на тему: **Стеганографічний захист аудіо файлів**

Виконав: студент V курсу, групи КН-51м
спеціальності
122 – "Комп'ютерні науки"
(шифр і назва напряму підготовки, спеціальності)
Глушко В.А.
(прізвище та ініціали)
Керівник Різник О.Я.
(прізвище та ініціали)
Рецензент _____
(прізвище та ініціали)

Львів – 2022 р.

Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)

ННІ деревообробних та комп'ютерних технологій і дизайну
Кафедра інформаційних технологій
Рівень вищої освіти другий (магістерський)
Спеціальність 122 "Комп'ютерні науки"
(шифр і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____Крошний І. М.
"____"
_____2022 року

З А В Д А Н Н Я
НА ДИПЛОМНУ РОБОТУ СТУДЕНТУ

Глушко В.А.
(прізвище, ім'я, по батькові)

1. Тема роботи **Стеганографічний захист аудіо файлів**

керівник роботи, канд. техн. наук, доцент Різник Олег Яремович.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від 13.09.2022 року № С-618
2. Термін подання студентом роботи 19. 12. 2022 р.
3. Вихідні дані до роботи:
- провести огляд стеганографічних алгоритмів захисту аудіо файлів
- дослідити математичну модель стеганографічного захисту аудіо файлів;
- розробити на мові програмування С++ програмний продукт з інтуїтивним інтерфейсом;
- провести тестування роботи розробленого програмного продукту.

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)
Розділ 1. Стан проблемної області
Розділ 2. Інформаційне забезпечення
Розділ 3. Математичне забезпечення
Розділ 4. Програмне забезпечення
5. Перелік графічного матеріалу (системний аналіз, розробка та відображення алгоритмів роботи стеганографічних методів захисту аудіо файлів, структура програмного рішення, експериментальна частина)
Додаток А. Вихідний код розробленого програмного забезпечення
6. Дата видачі завдання 01 вересня 2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної роботи	Строк виконання етапів роботи	Примітка
1	Огляд літературних джерел	01.09-30.09.2022	виконано
2	Розділ 1. Стан проблемної області	01.10-15.10.2022	виконано
3	Розділ 2. Інформаційне забезпечення	16.10-31.10.2022	виконано
4	Розділ 3. Математичне забезпечення	01.11-15.11.2022	виконано
5	Розділ 4. Програмне забезпечення	16.11-30.11.2022	виконано
7	Експериментальна частина	01.12-09.12.2022	виконано
8	Оформлення дипломної роботи	10.12-14.12.2022	виконано
9	Підготовка до захисту дипломної роботи, оформлення презентації	15.05-17.12. 2022	виконано

Студент _____ Глушко В.А.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Різник О.Я.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Магістерська робота містить 73 сторінки пояснень, 24 рисунки, 3 таблиці, 1 додаток, 16 посилань.

Метою роботи є розгляд проблем стеганографії та стегоаналізу аудіофайлів.

Основні завдання: розгляд функції приховування та виявлення прихованих вкладень в аудіофайлах, створення програми для приховування даних у аудіофайлах, пропозиція методів виявлення прихованих даних у аудіофайлах.

Розроблений додаток дозволяє приховувати дані в аудіофайлах, а також визначати наявність стего вкладень в певних типах аудіофайлів.

Ключові слова: стеганографія, стеганоаналіз, аудіо файли, багатотомність, методи стискування, статистичні тести.

ANNOTATION

The master's thesis contains 73 pages of explanations, 24 figures, 3 tables, 1 appendix, 16 references.

The purpose of the work is to consider the problems of steganography and stegoanalysis of audio files.

Main tasks: examination of the function of hiding and detection of hidden attachments in audio files, creation of a program for hiding data in audio files, proposal of methods of detection of hidden data in audio files.

The developed application allows you to hide data in audio files, as well as to determine the presence of stego attachments in certain types of audio files.

Keywords: audio files, compression methods, multivolume, statistical tests, steganoanalysis, steganography.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ

НЗБ	найменш значні біти
ПЗ	програмне забезпечення
ЦВЗ	цифровий водяний знак
ЦОЗ	цифрова обробка зображення
LSB	Least Significant Bit, найменьший значащий біт
PCM	pulse code modulation, імпульсно-кодова модуляція

ВСТУП

Тому в даний час професіонали повинні мати навички, які можна використовувати для захисту комп'ютерних даних, і розуміти проблему захисту інформаційних активів.

Необхідність реалізації заходів криптографічного захисту інформації дозволить зберегти її, а також протистояти її знищенню, розкраданню чи спотворенню.

Предметом дослідження вивчення алгоритмів шифрування аудіофайлів.

Метою дослідження є розробка алгоритму захисту інформації в аудіофайлах та розробка рекомендацій щодо забезпечення захисту аудіофайлів, що передаються.

Об'єктом дослідження є аудіо файли в інформаційниї комунікаційних мережах.

Предметом дослідження є вивчення алгоритмів шифрування аудіо файлів.

Задачі дослідження. Для вирішення проблем захисту аудіофайлsd необхідно врахувати вимоги до стеганосистем, які приховують інформацію в аудіосигналах:

- інформація повинна бути стійкою до спотворень;
- інформація не повинна спотворювати аудіосигнал;
- спроба видалити інформацію приводить до пошкодження аудіосигналу;
- інформація не повинна викликати чутливі зміни аудіосигналу.

Практична цінність роботи полягає в можливості захисту аудіофайлів від неналежного використання.

РОЗДІЛ 1. СТАН ПРОБЛЕМНОЇ ОБЛАСТІ

Для захисту інформації її передають за допомогою криптографії — науки про методи захисту, конфіденційність та достовірність інформації.

Довгий час методи захисту розроблялися виключно державними органами, які реалізація вважалася винятковим правом держави. Проте останніми роками з розвитком комерційної та підприємницької діяльності збільшилася кількість спроб отримання неавторизованого доступу до інформації.

1.1. Класифікація шифросистем

Складність найкращого алгоритму вирішення проблеми розкриття ключа за наявності відкритого тексту та відповідного зашифрованого тексту. Складність найкращого алгоритму вирішення проблеми розкриття ключа з використанням правильно підібраних пар шифрів та відповідних відкритих текстів і т. д.

Ці та інші терміни у криптографії часто використовуються для розрізнення сильних та слабких криптографічних систем. Тобто, якщо криптосистема дозволяє розкрити ключ за певних умов, то така система вважається слабкою, інакше стійкою. Зрозуміло, що у умовах відкриття ключа має відбуватися за прийнятний час.

Схема процесу передачі інформації наведена на рис. 1.1.

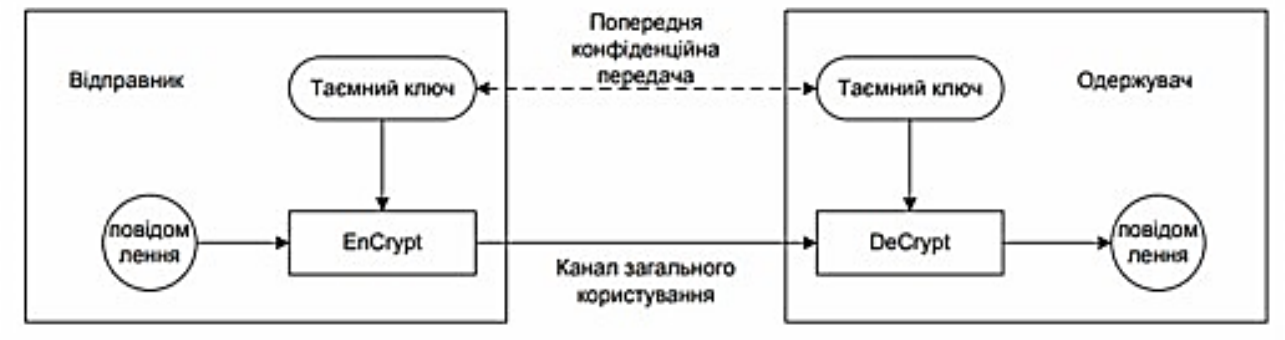

Рис. 1.1. Схема процесу передачі інформації

Безпека даних включає цілісність, доступність і конфіденційність даних.

1.2. Методи стеганографії

На рис. 1.2. накведені області застосування стеганографії.

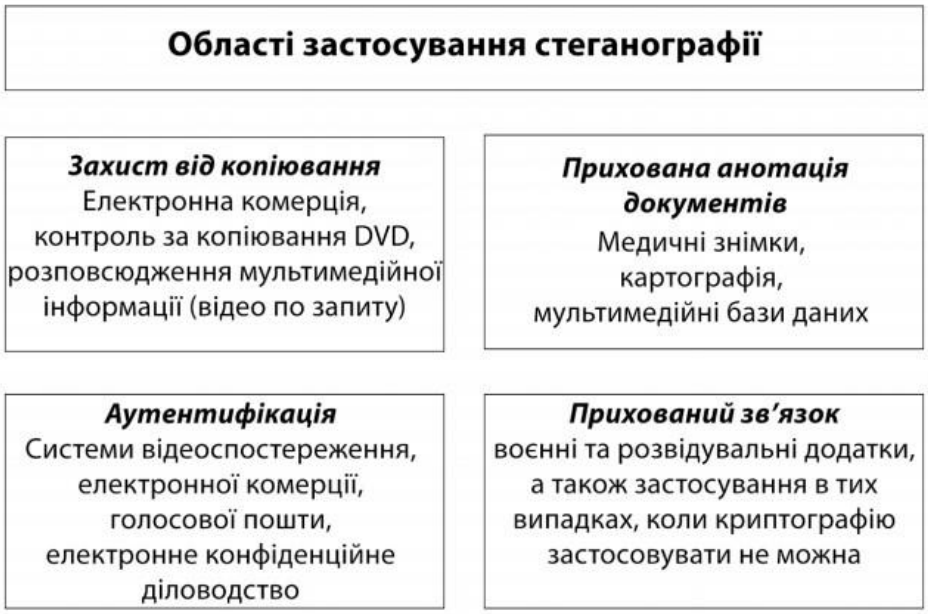

Рис. 1.2. Класифікація областей застосування стеганографії

Використання криптографії дозволяє приховати зміст конфіденційної інформації, але може приховати сам факт їх наявності чи передачі. Стеганографічні методи, спрямовані на приховування самого факту наявності конфіденційної інформації. Як інформацію можна використовувати: текст, повідомлення, зображення тощо.

РОЗДІЛ 2. ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ

2.1. Методи криптографії

Криптографія є одним із основних інструментів забезпечення конфіденційності та цілісності інформації, авторизації, електронних платежів, оперативного контролю процесів управління та обробки даних. Найбільш поширеним є захист даних у комп'ютерних мережах. Зазвичай це локальні корпоративні мережі, підключені до Інтернету.

Існує класифікація криптографічних алгоритмів, що представлені на рис. 2.1.

Рис. 2.1. Класифікація криптографічних алгоритмів

Криптографічна схема такої системи, яка потребує шифрування інформації, представлена на рис. 2.2.

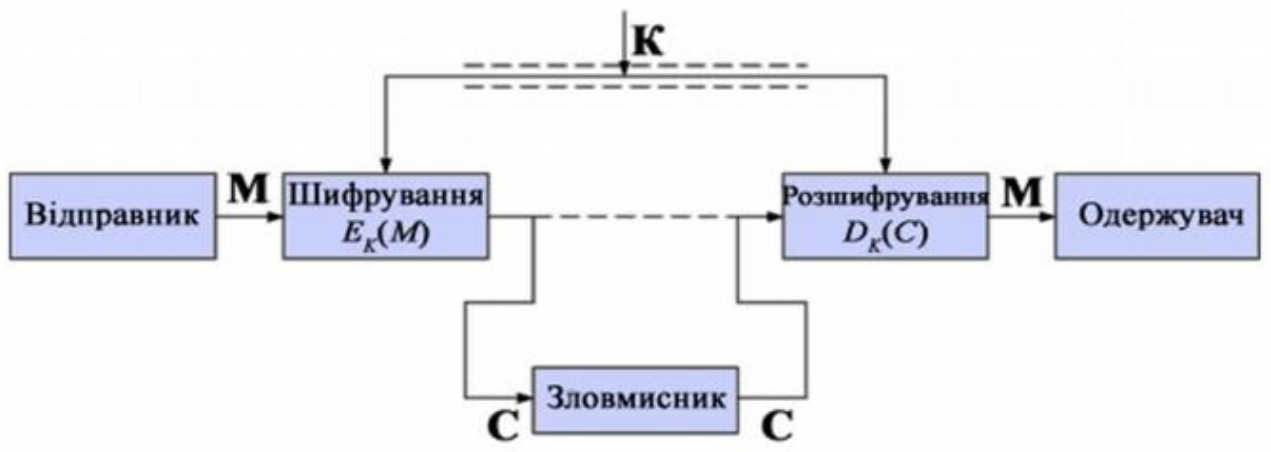

Рис. 2.2. Схема криптосистеми

Якщо той самий ключ використовується і одержувачем, і відправником інформації, то така система є симетричною.

2.1.1. Симетричне шифрування

Алгоритми шифрування та дешифрування даних набули широкого поширення в обчислювальній техніці.

Схема алгоритму симетричного шифрування показана на рис. 2.3.

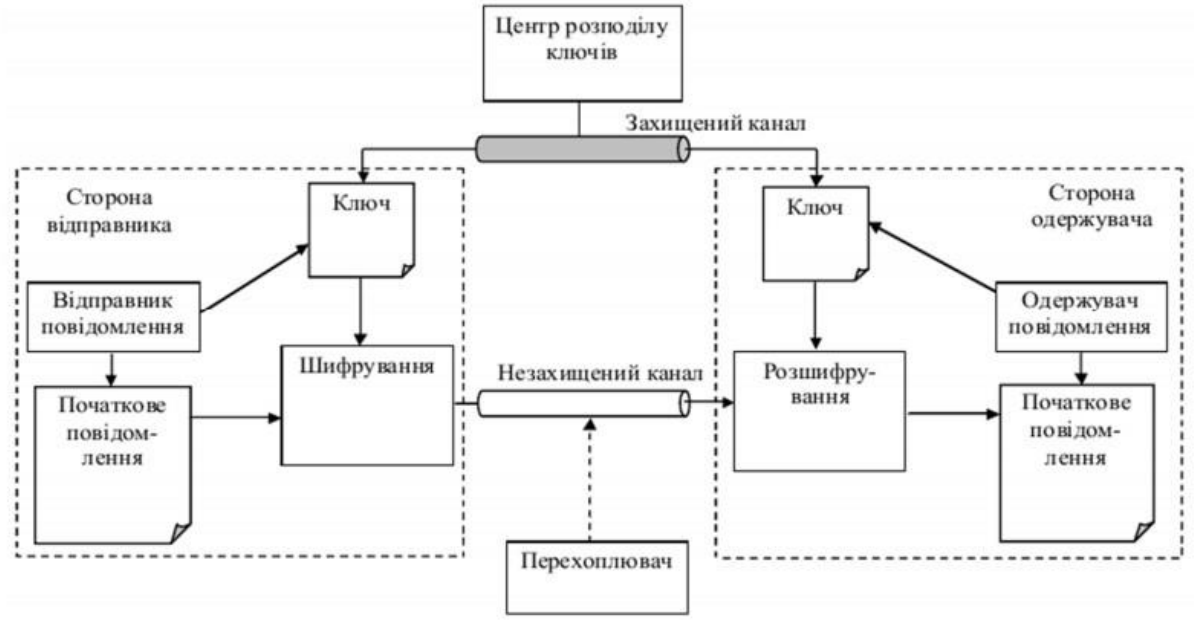

Рис. 2.3. Алгоритм симетричного шифрування

Схема підстановки в алгоритмі AES з використанням S-боксів наведена на рис. 2.4.

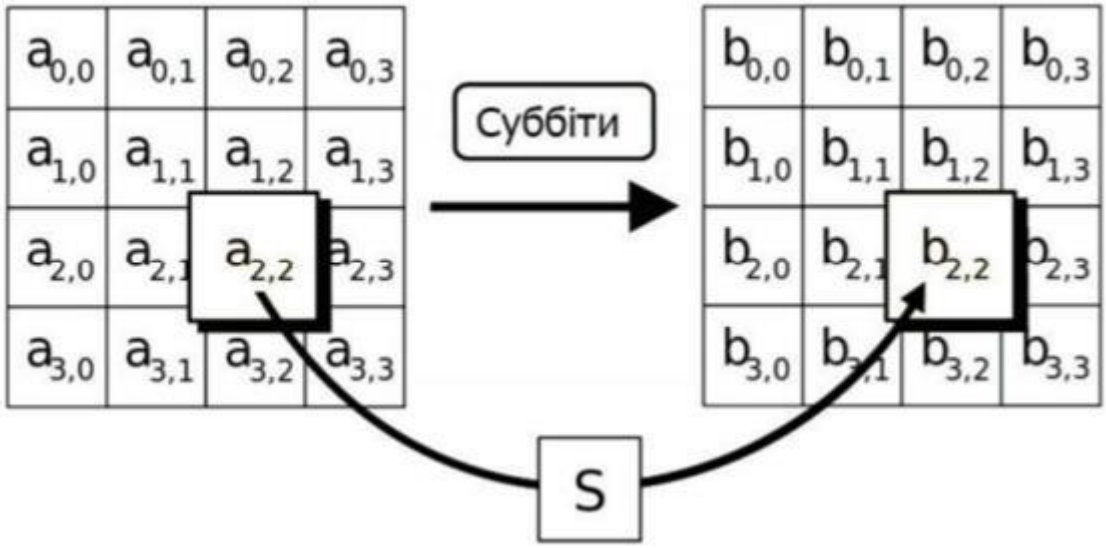

Рис. 2.4. Схема підстановки в алгоритмі AES з використанням S-боксів

2.1.2. Асиметричне шифрування

Цей тип шифрування інформації називається шифруванням інфраструктури відкритих ключів (PKI). Тому тут при шифруванні та розшифруванні інформації використовуються різні ключі – відкриті та закриті.

Перший ключ відкритий чи публічний. За його назвою можна зрозуміти, що він може передавати його іншим користувачам. Треба ділитися. Адже відкритий ключ використовується для перевірки цифрових підписів та шифрування повідомлень.

Закритий ключ — це особистий ключ. Це не дозволяє всім користувачам мережі дізнатися ключ шифрування.

На рис. 2.5. наведена схема асиметричнлгое шифрування

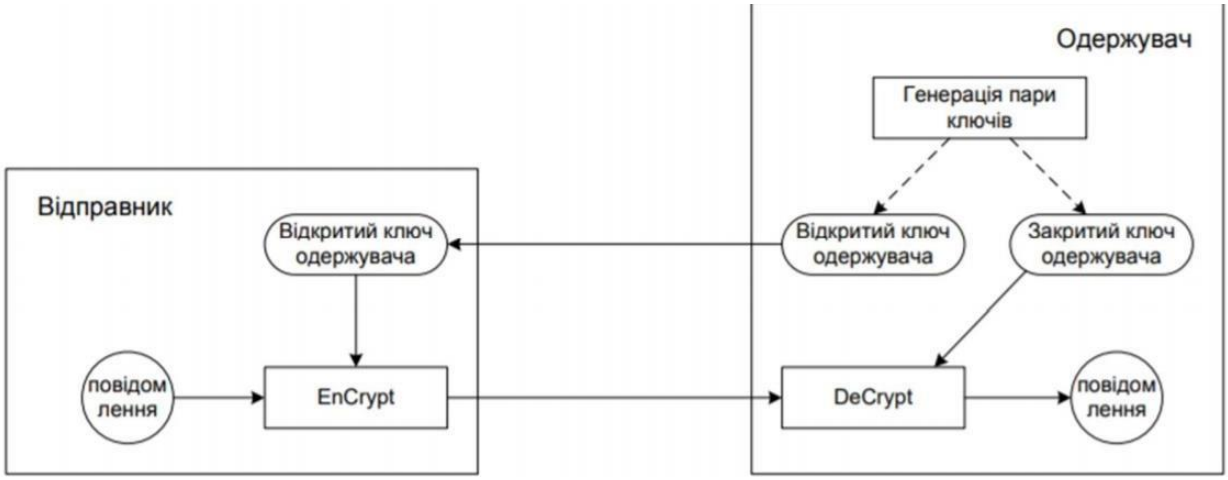

Рис. 2.5. Асиметричне шифрування

2.1.3. Алгоритми шифрування

Криптосистема DES (стандарт шифрування даних) є гарним прикладом криптографічного алгоритму, розробленого відповідно до розповсюдження та міграції.

Цей алгоритм шифрування у вигляді блок-схеми:

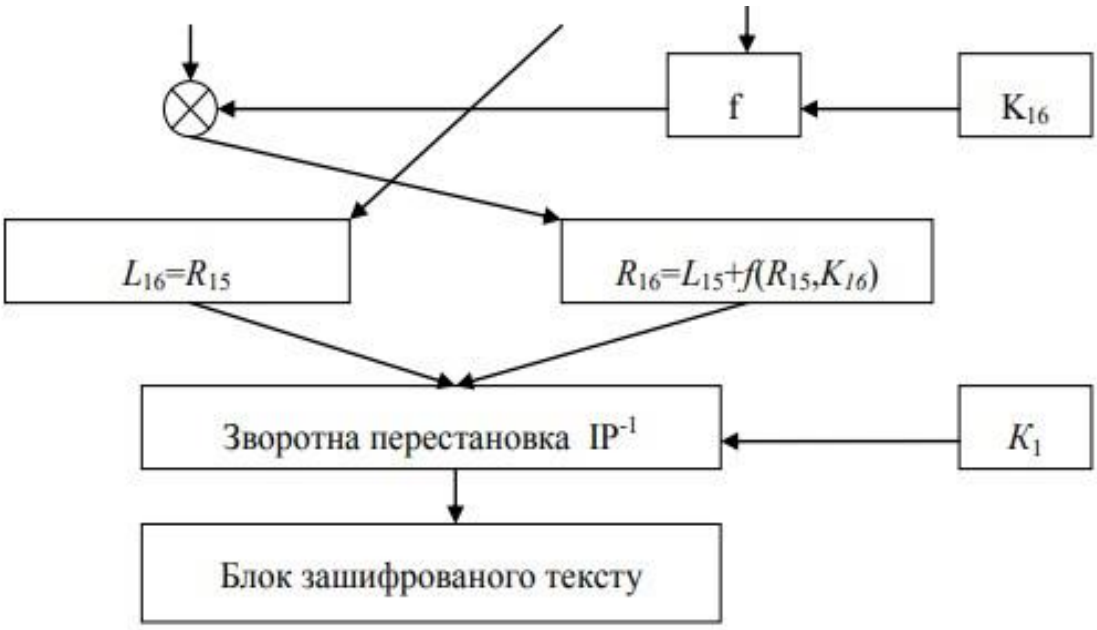

Рис. 2.6. Блок-схема криптосистеми шифрування DES

Перші відгуки фахівців щодо алгоритму також були виключно позитивними. Зокрема, було зазначено, що цей алгоритм є надзвичайно криптостійким.

Інший тип криптосистеми – прозорий. Цей алгоритм, що працює у фоновому режимі як драйвер фільтра, шифрує та розшифровує дані без втручання користувача та відстежує всі звернення.

Функція прозорого шифрування забезпечує швидку та зручну роботу із секретними даними для кількох користувачів одночасно.

Якщо користувачів кілька, а сам зашифрований файл знаходиться не на комп'ютері, а на віддаленому сервері, у кожного користувача має бути своя унікальна пара особистих ключів для цього зашифрованого файлу.

На рис. 2.7. наведена схема прозорого шифрування.

Рис. 2.7. Схема прозорого шифрування

2.1.4. Метод ЕЦП

Метою автентифікації є захист від можливих шкідливих атак. Ось деякі типи можливих атак:

- маскування;
- заміна;
- активне захоплення;
- повторити.

Такі шкідливі дії можуть завдати істотних збитків банківським та комерційним структурам, державним підприємствам тощо.

Документ, що підтверджує належність відкритого ключа підписувачу, надається центром сертифікації і називається сертифікатом відкритого ключа.

При зміні одного з цих елементів перевірити автентичність цифрового підпису стає неможливо. Таким чином, електронний документ набуває юридичної значущості завдяки електронному цифровому підпису.

В ЕЦП є дві основні процедури:

- процедура цифрового підпису;
- процедура перевірки електронного підпису.

Електронні підписи можуть використовуватися для проведення фінансових транзакцій, а також часто використовуються для розповсюдження програмного забезпечення, складання бюджетних та податкових звітів та виявлення підробок.

2.2. Висновки

У першій частині розглядаються методи шифрування інформації. Якісні симетричні алгоритми швидші за якісні асиметричні алгоритми; Недоліком існуючих симетричних алгоритмів є наявність секретного ключа обох сторонах передачі. Надається основна інформація про алгоритми шифрування.

РОЗДІЛ 3. МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1. Створення сертифікату в програмі CyberSafe

Щоб створити сертифікат користувача, перейдіть до відповідного розділу: Сертифікати та ключі → Особисті ключі → Створити.

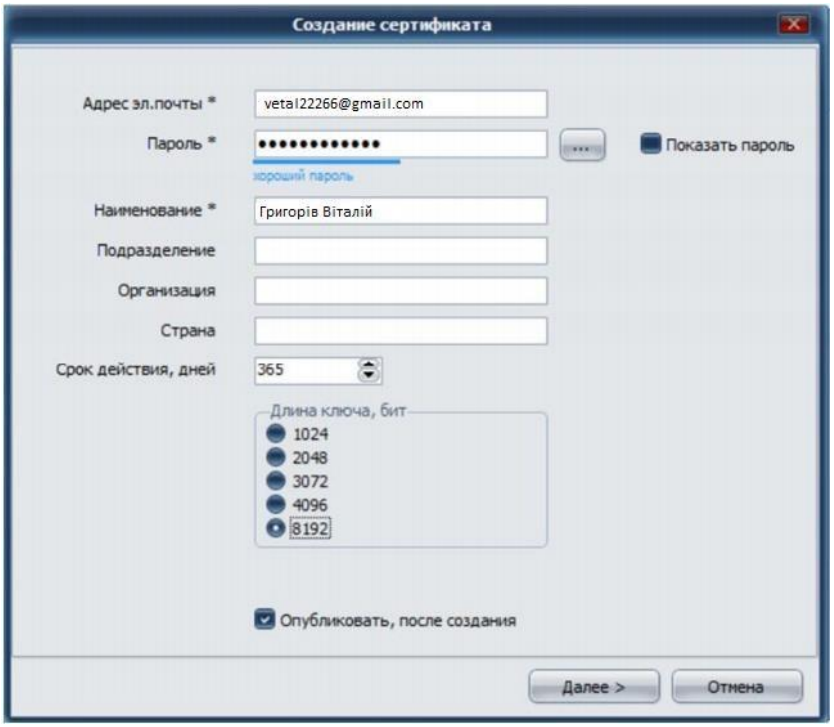

Рис. 3.1. Створення сертифікату

Обов'язкові поля відзначені *. У полі Адреса електронної пошти необхідно ввести свою діючу адресу електронної пошти, куди прийде код для сертифіката програми CyberSafe.

Програма згенерує закритий ключ сертифіката користувача, який захищений паролем, який буде надіслано на електронну пошту; цей пароль необхідний для прозорого шифрування, або при необхідності треба буде експортувати закритий ключ до окремого файлу.

Далі слідує процес генерації ключів сертифікатів, після чого на вказану вами адресу електронної пошти буде відправлено код підтвердження, який слід ввести в поле, показане на рис. 3.2. Потім ваш сертифікат буде опубліковано на сервері програм.

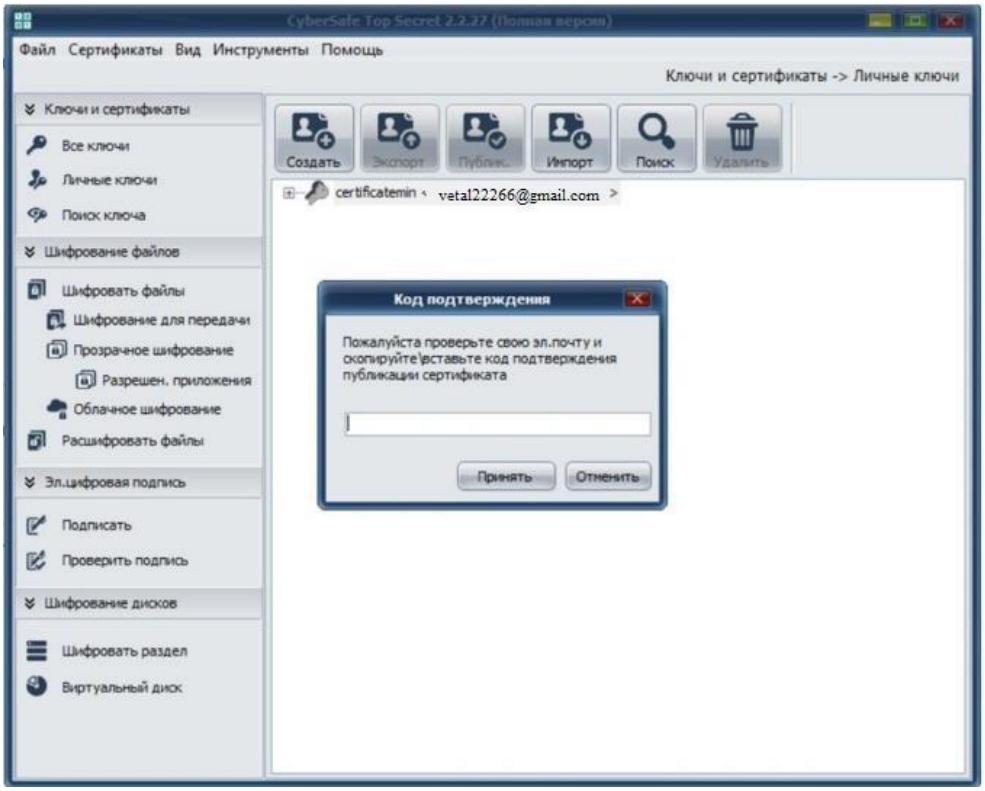

Рис. 3.2. Підтверджена публікація сертифікату

Після підтвердження пароля процедура публікації сертифіката вважається завершеною і результат представлений на рис. 3.3.

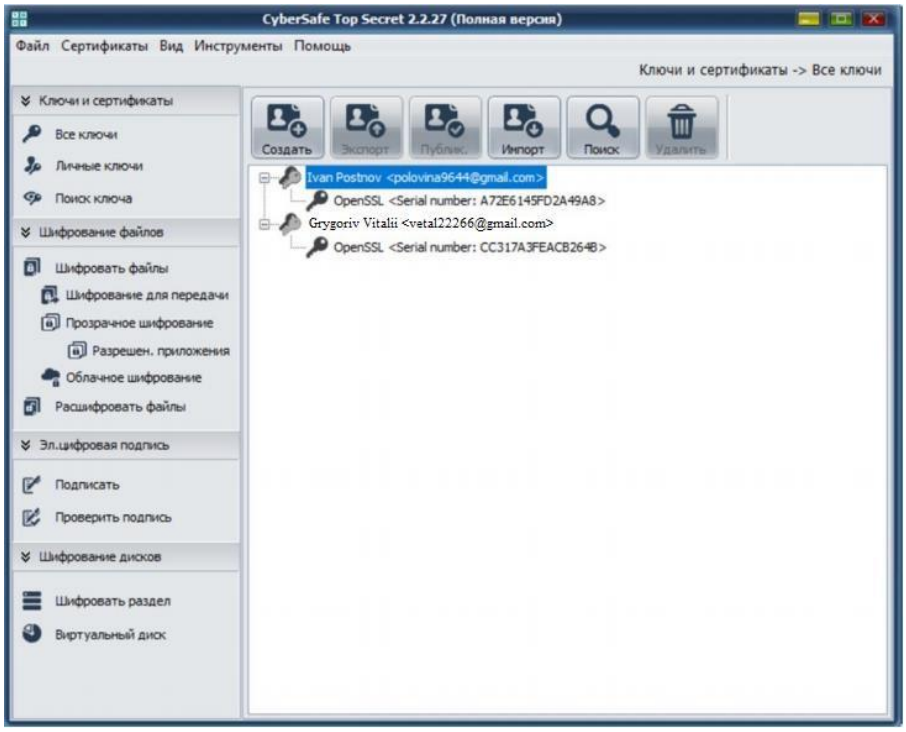

Рис. 3.3. Опубліковані сертифікати

На час створення сертифіката з більшою довжиною ключа (192 біти) витрачено 1 хвилину 05 секунд, з більш короткою довжиною ключа (124 біти) це було 35 секунд.

3.3. Алгоритми шифрування

3.3.1. Шифрування файлів

Шифрування часто використовується для захисту файлів від несанкціонованого доступу. Шифрування файлів стане в нагоді: для конфіденційного зберігання на комп'ютері або для пересилання іншим користувачам.

За допомогою CyberSafe ви можете зашифрувати файли для передачі іншим користувачам трьома способами: за допомогою шифрування паролів, на основі сертифікатів (ключів) або шляхом створення зашифрованого ZIP-архіву, що саморозпаковується.

Якщо ви хочете зашифрувати файли для подальшого надсилання іншим користувачам, вони будуть зашифровані з використанням відкритих ключів користувачів, яким ви надсилаєте файли. Для виконання цих кроків у вас має бути посилання на сертифікат користувача CyberSafe. Якщо ви не маєте сертифіката користувача, ви можете використовувати функцію пошуку для знаходження його на сервері відкритих ключів CyberSafe, використовуючи електрону пошту користувача.

Також можна виділити дві важливі особливості програми: система довірених додатків та двофакторна автентифікація. У налаштуваннях програми можна встановити двофакторну автентифікацію або автентифікацію за паролем.

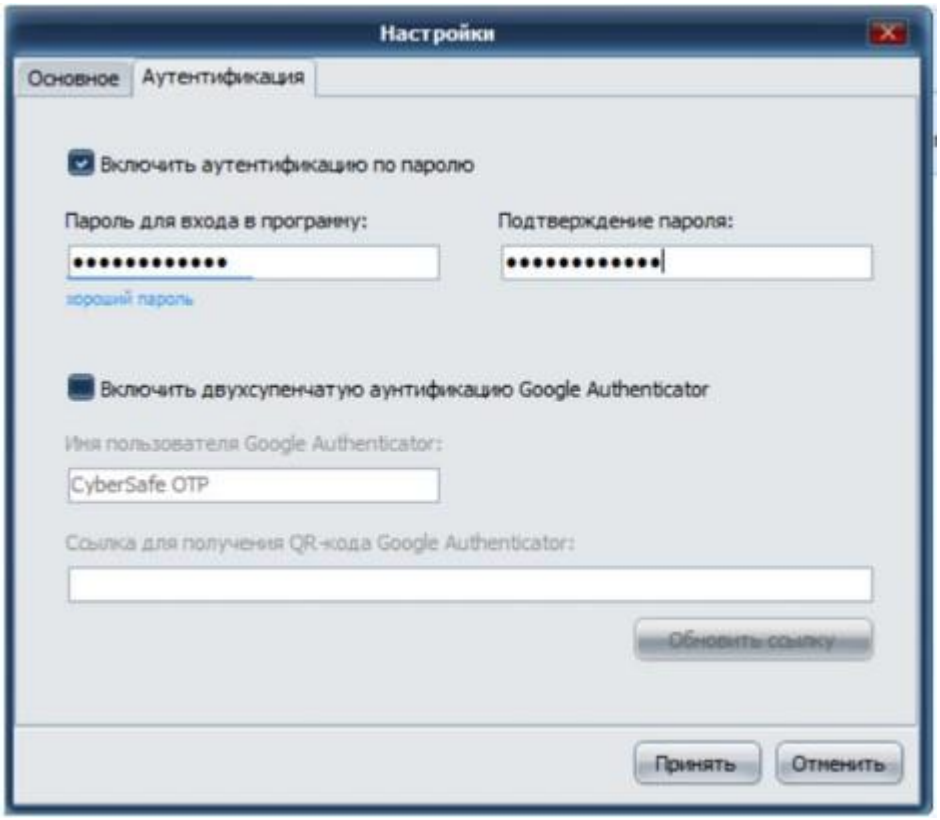

Рис. 3.4. Автентифікація користувача

3.3.2. Шифрування на основі сертифікатів

Цей тип шифрування часто використовується:

- коли ви не повідомляєте пароль до файлу;
- забезпечення найвищого рівня захисту файлів;
- шифрувати файли для подальшого використання, а також вести обмін файлами з користувачами, у яких є програма CyberSafe.

Щоб зашифрувати файл для подальшої безпечної передачі, користувач вибирає: *Шифрування файлу → Шифрування для передачі одержувачам*:

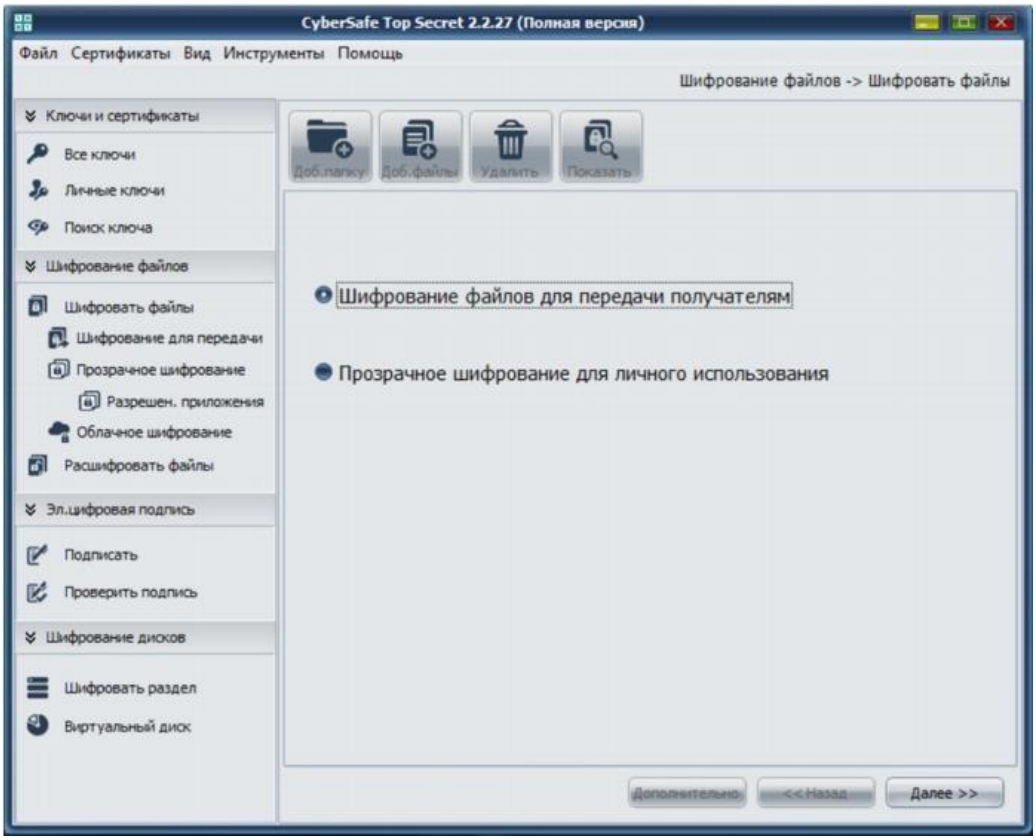

Рис. 3.5. Вкладка шифрування файлів

Потім додаємо файл для шифрування (рис. 3.6).

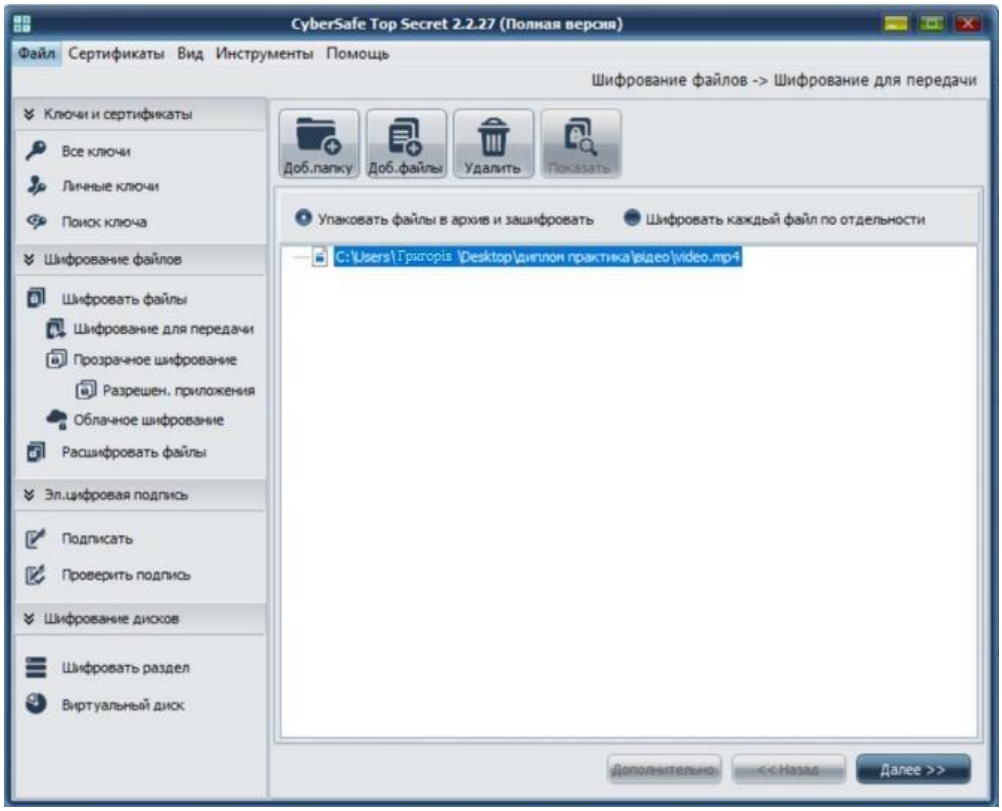

Рис. 3.6. Обраний файл

Далі переходимо до *Зашифрувати файли ключами одержувачів* та *Далі:*

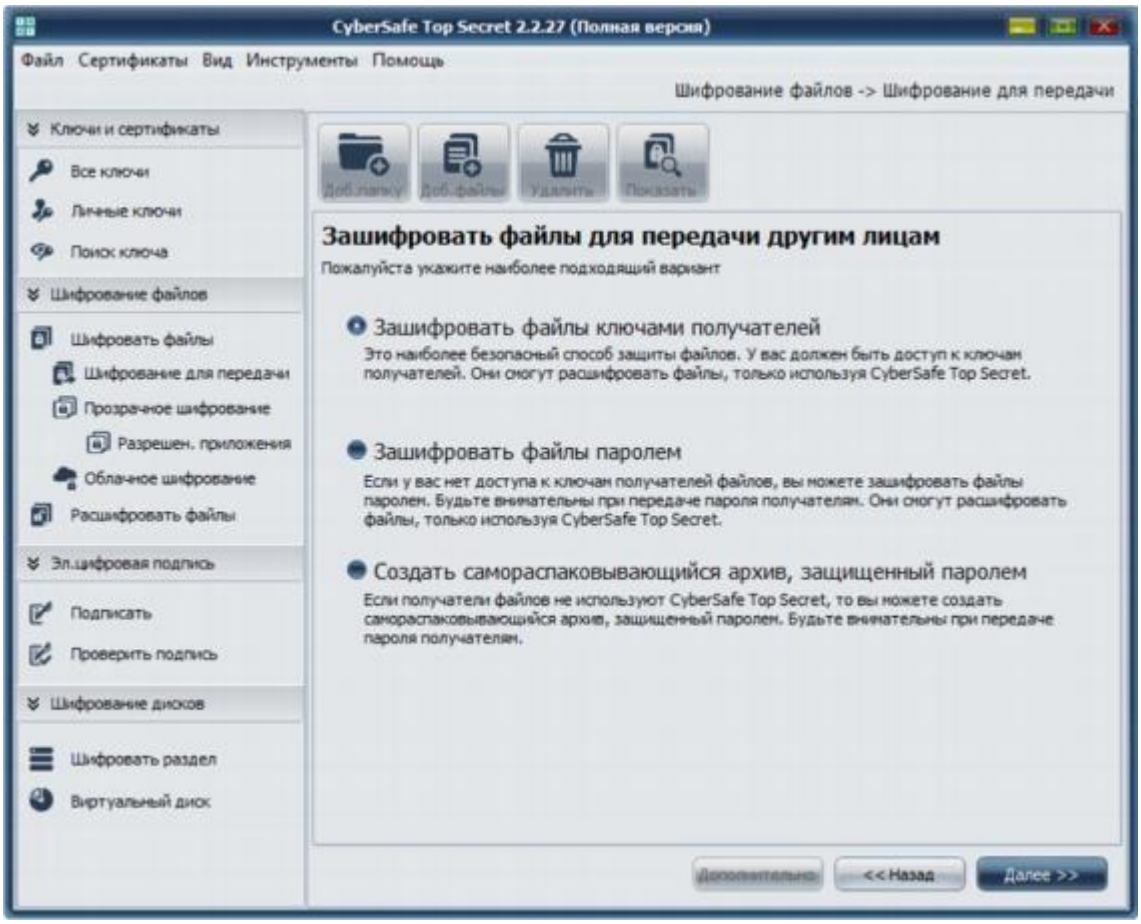

Рис. 3.7. Вибір варіантів шифрування

Потім зі списку беремо користувачів з ключами, які ми використовуватимемо для шифрування файлів.

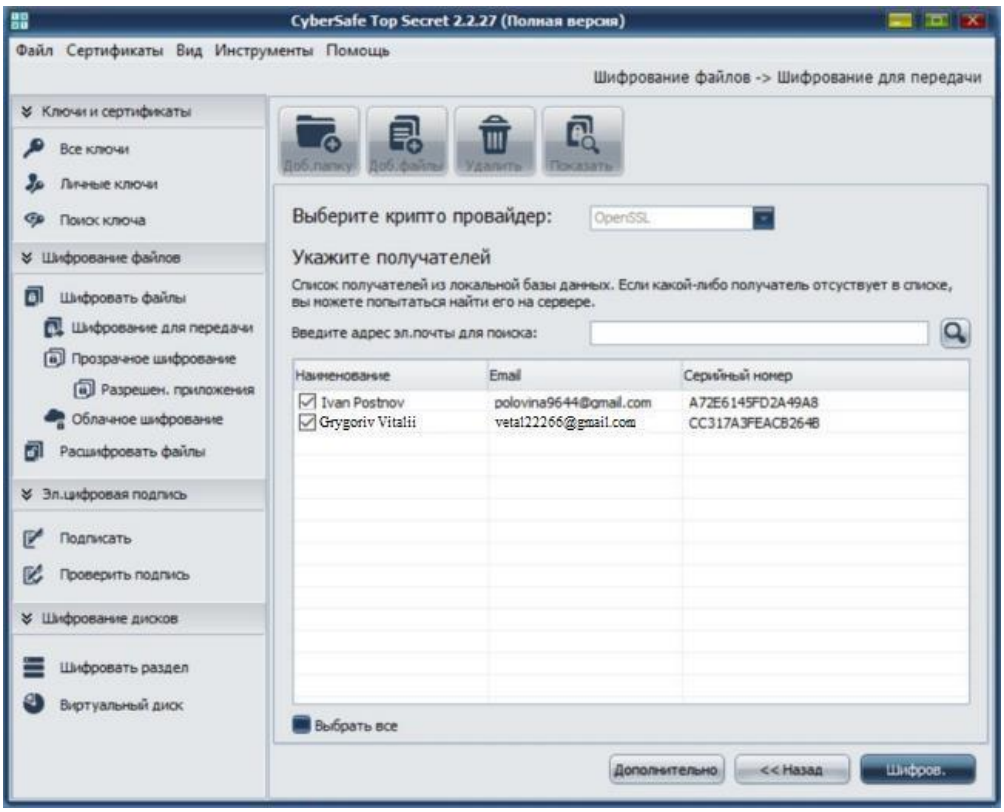

Рис. 3.8. Список активних доступних користувачів

Перед використанням кнопки *Шифрувати*, якщо є вибір методу шифрування, ви можете використовувати додаткові функції у вікні додаткових налаштувань, натиснувши *Додатково*:

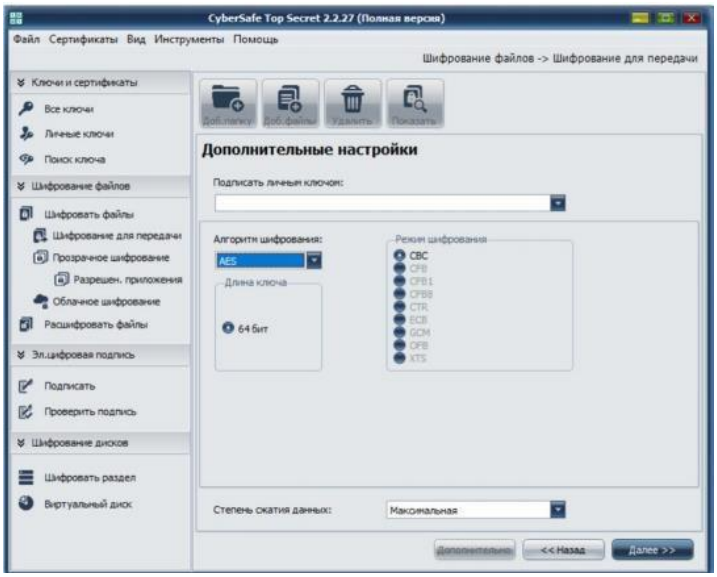

Рис. 3.9. Вікно додаткових налаштувань

У цьому вікні можливо:

- вкажіть, наскільки сильно хочете стиснути файл;
- поміеяти алгоритм шифрування за умовчанням та довжину ключа;
- вибрати закритий ключ для цифрового підпису файлу (у разі шифрування з відкритим ключем).

Процес шифрування почнеться відразу після натискання кнопки *Шифрувати*. Наприкінці процедури з'явиться таке вікно:

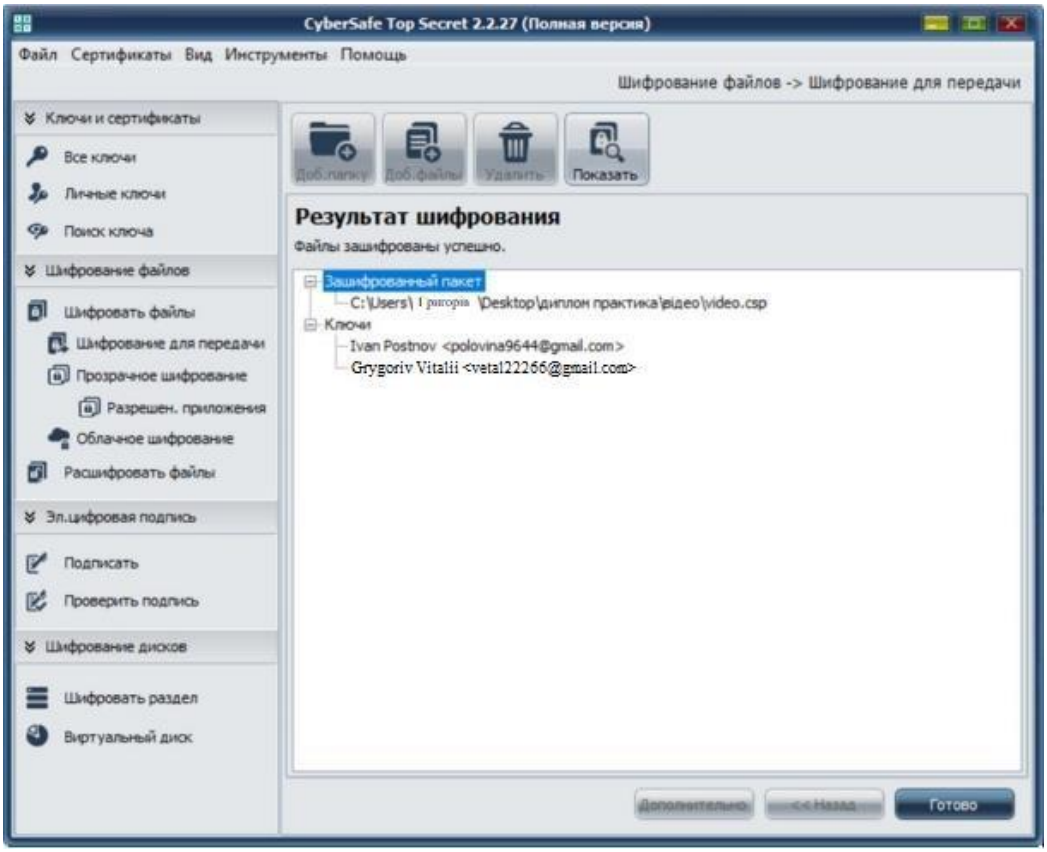

Рис. 3.10. Результат шифрування

Після того, як необхідні файли будуть зашифровані, їх можна надіслати іншим користувачам. Отримавши файли, користувач розшифровує їх за допомогою CyberSafe та свого закритого ключа. Файли можуть бути розшифровані всіма одержувачами, відкриті ключі яких використовувалися при шифруванні. У результаті кожен користувач отримує той самий файл.

Якщо ви відправляєте файл другому користувачеві, зашифруйте його з використанням відкритих ключів - найкраще рішення, яке зазвичай вибирають

коли потрібний найвищий рівень безпеки. Цей метод шифрування дозволяє шифрувати дані за допомогою асиметричного ключа завдовжки до 256 біт.

3.3.3. ЕЦП – електроний цифровий підпис

Якщо необхідно створити цифровий підпис, перейдемо на *Електронний цифровий підпис → Підпис*. При виборі файлу з'явиться опція *шифрування файлів ключами одержувачів*.

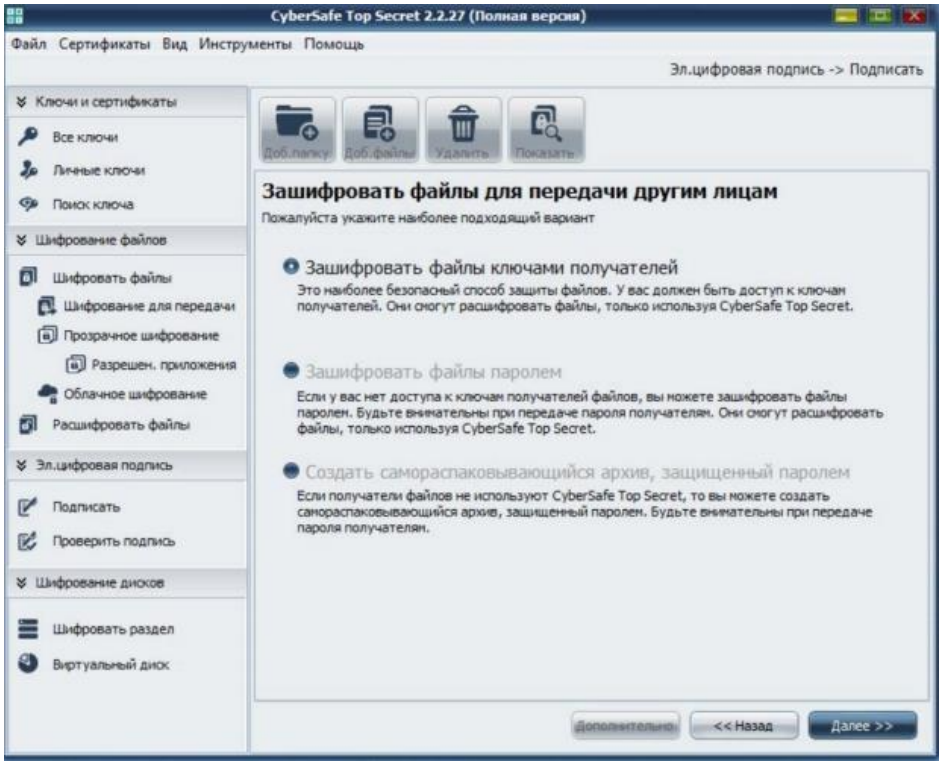

Рис. 3.11. ЕЦП

Після вибору ключів одержувачів необхідно вибрати закритий ключ для ЕЦП файлу, алгоритми шифрування та довжину ключа.

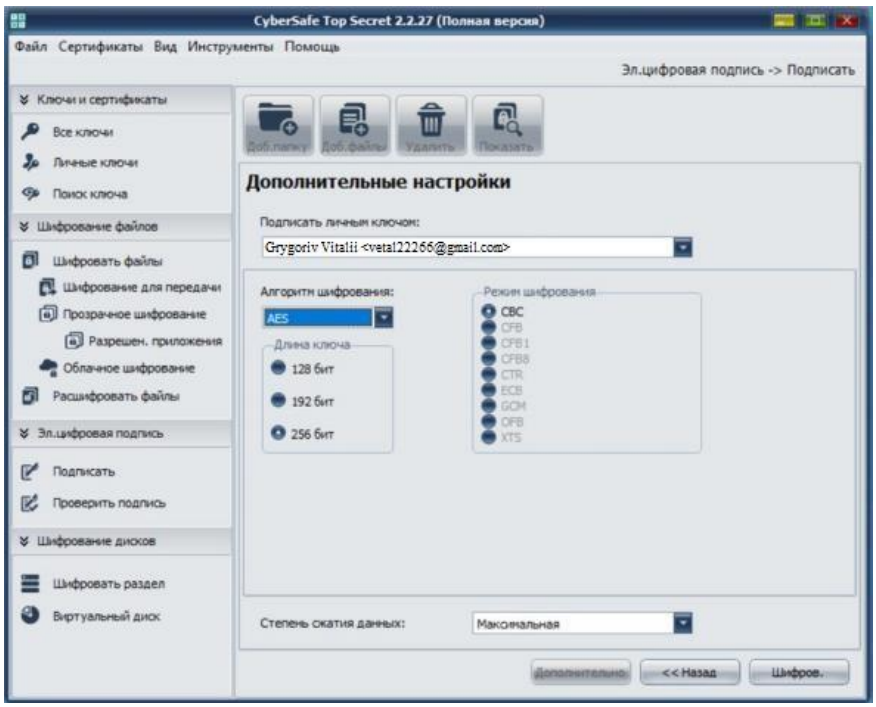

Рис. 3.12. Меню за типом шифрування

Перед шифруванням файлу паролем сертифіката перевірте електронний цифровий підпис. Після підтвердження пароля автоматично розпочнеться шифрування файлів.

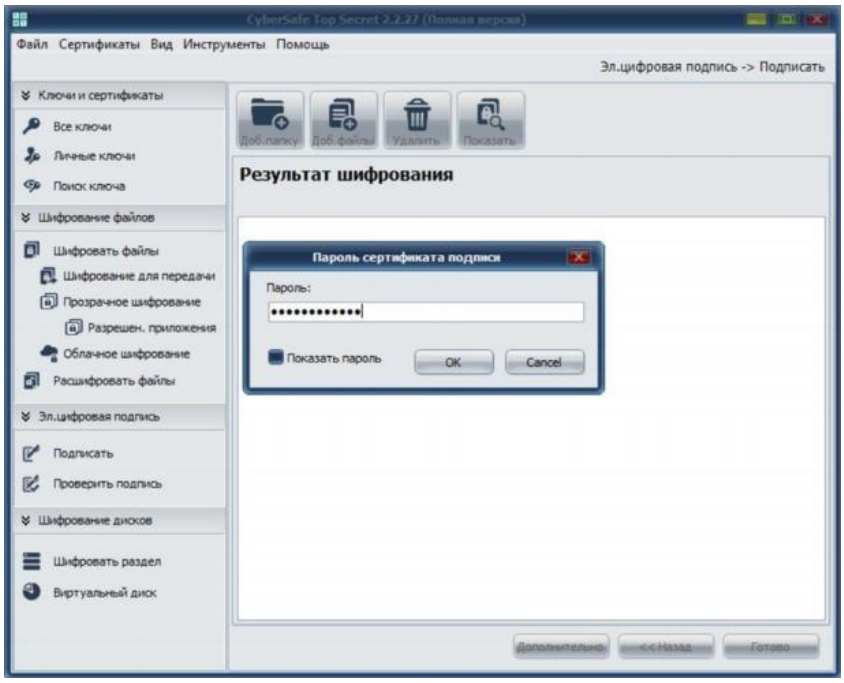

Рис. 3.13. Підтвердження ЕЦП

Після закінчення шифрування у меню «Зашифрований пакет» з'явиться шлях до зашифрованих файлів. Розділ «Ключі» міститиме користувачів, чиї відкриті ключі використовуються для шифрування.

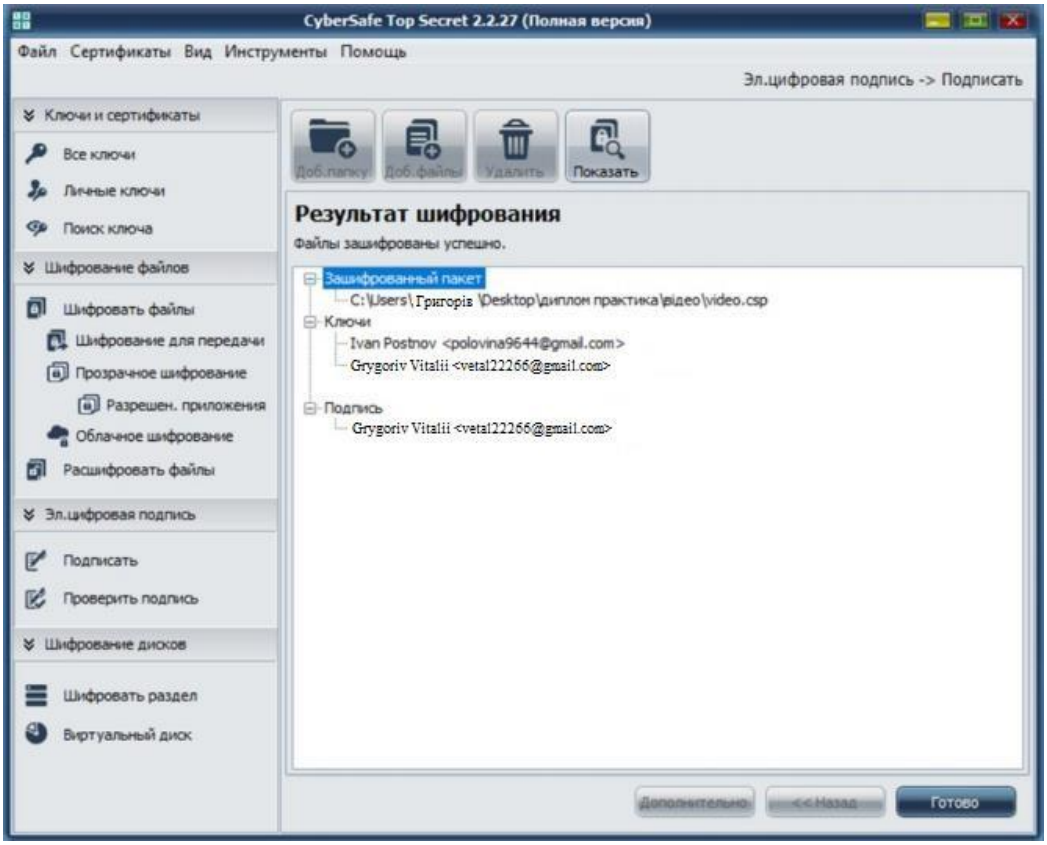

Рис. 3.14. Результат даного типу шифрування

3.4. Шифрування файлів

Шифрування даних допомагає зберегти їх конфіденційність, але водночас привертає надто багато уваги.

В даний час активно набирають популярності комбінаторні методи, які часто використовуються для декодування та кодування різних видів інформації.

Стеганографія дозволяє без особистих зусиль та спеціальних знань приховати файл у графічному, текстовому чи звуковому форматі.

3.4.1. Процес шифрування графічного зображення

Для шифрування зображення виберемо ключ довжиною 128 біт, алгоритм шифрування AES та максимальний рівень стиснення. Для цифрового підпису ми використовуємо закритий ключ мінімальна довжина якого становить 128 біт.

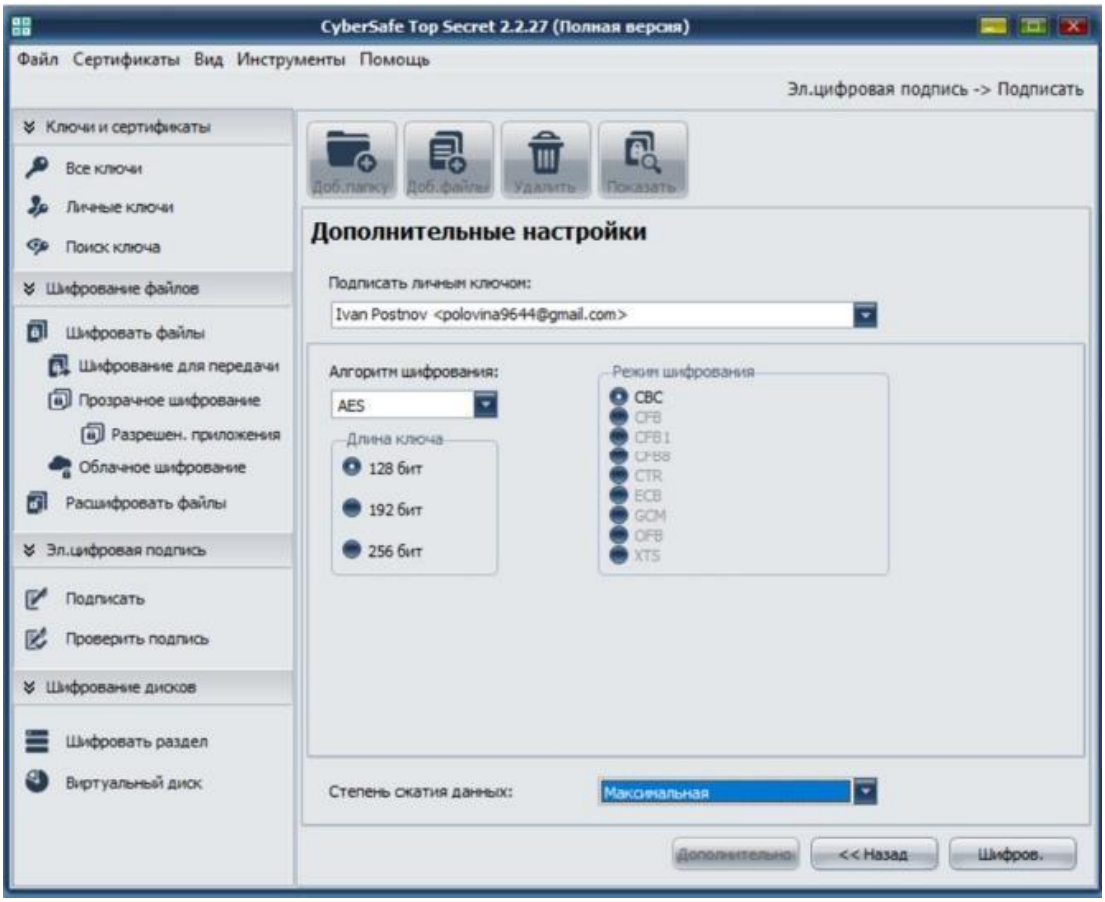

Рис. 3.15. Параметри шифрування

Файл має розмір 2,2 МБ. Час шифрування графічного зображення – 7,16 секунди.

Результати цього шифрування показані нижче. На рис. 3.16 показано вигляд фотографії до шифрування, на рис. 3.17 показаний текстовий документ із текстовою інформацією про фотографію, що підлягає шифруванню.

Рис. 3.16. Зображення до шифрування

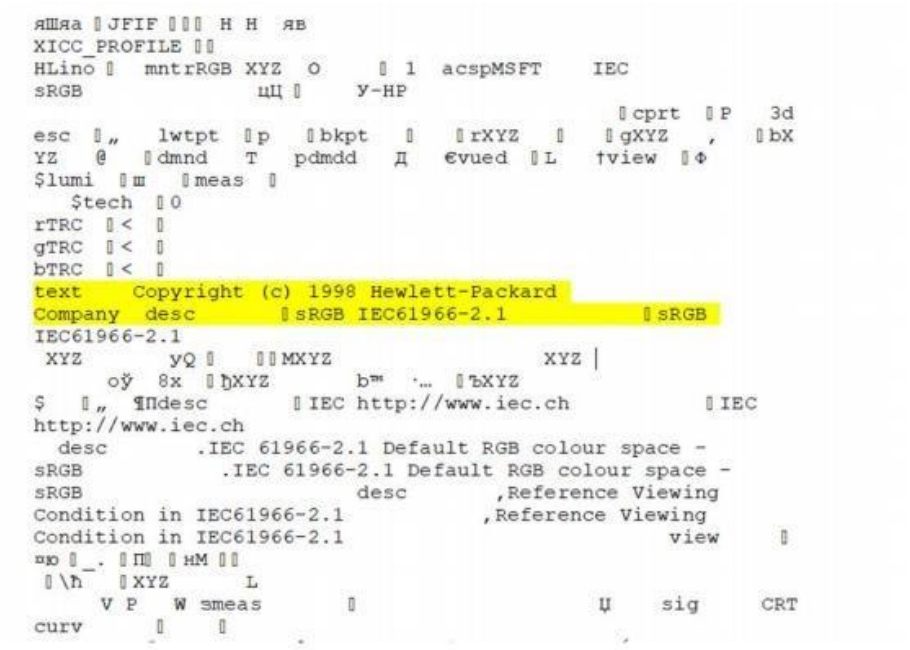

Рис. 3.17. Інформація про зображення перед шифруванням

Результат перевірки інформації про зашифроване зображення можна побачити на рис. 3.18.

Рис. 3.18. Результат шифрування

3.4.2. Процес шифрування звукового файлу

Пропонований метод шифрування аудіоінформації може бути використаний для різних цілей, у тому числі:

- приховування даних;
- приховування змісту актуальної інформації.

Для прикладу, виберемо аудіофайл з розширенням .flac та розміром 39 МБ. Відкрийте цей файл у текстовому редакторі.

На рис. 3.19 показаний аудіофайл до шифрування.

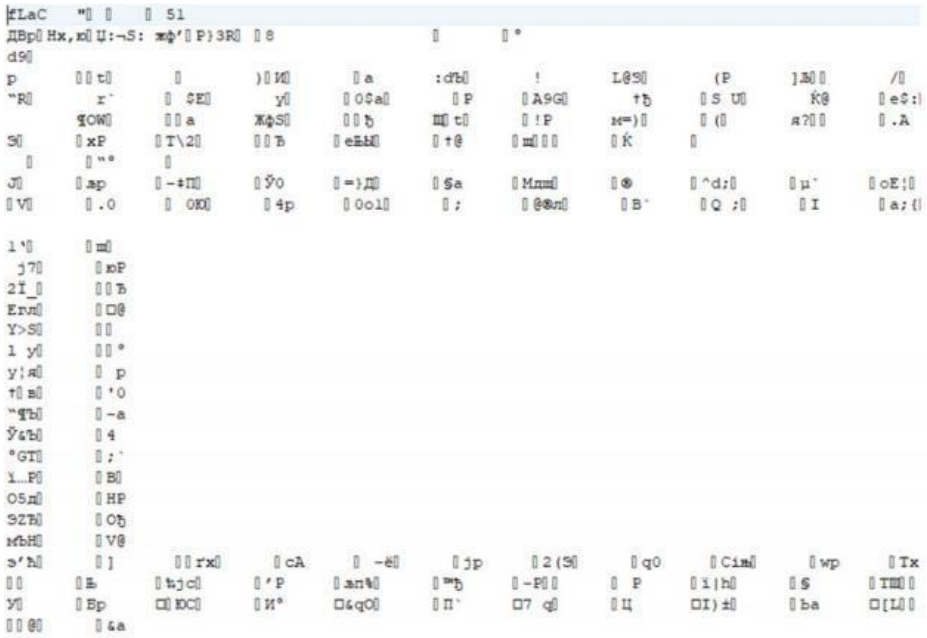

Рис. 3.19. Аудіофайл до шифрування

За допомогою CSTS шифруємо аудіофайл. Зашифруємо його за алгоритмом AES (128 біт) методом EDS із ключем мінімальною довжини. Час шифрування аудіофайлу становив 26,34 секунди. Відкрийте зашифрований файл у текстовому редакторі, щоб порівняти його з оригіналом. 3.20.

Рис. 3.20. Зашифрований файл

Після закінчення шифрування розмір файлу виріс на 53 кб, тому аудіофайл захищений ключем шифрування.

3.4.3. Процес шифрування відеофайлу

Для шифрування відеофайлу виберіть алгоритм шифрування DES, режим шифрування – SBS, довжина ключа шифрування – 64 біти, стиснення файлу – максимальне. Цей файл підписуємо своїм ключем, максимальна довжина якого 256 біт.

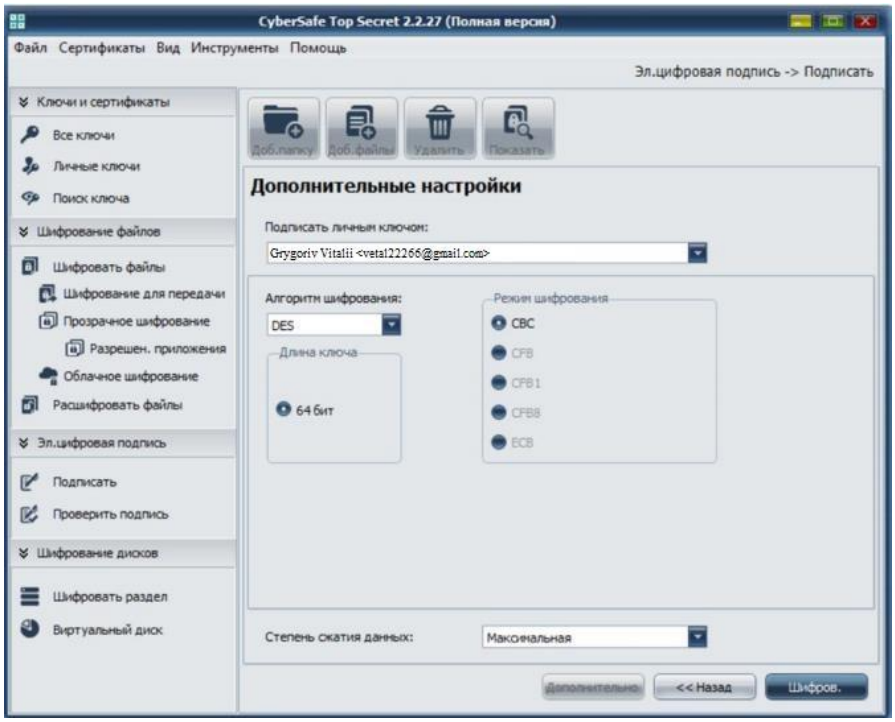

Рис. 3.21. Вікно з додатковими налаштуваннями

Файли успішно зашифровані. Тут шифрування зайняло 37,92 секунд. Дані після процедури подано на рис. 3.22.

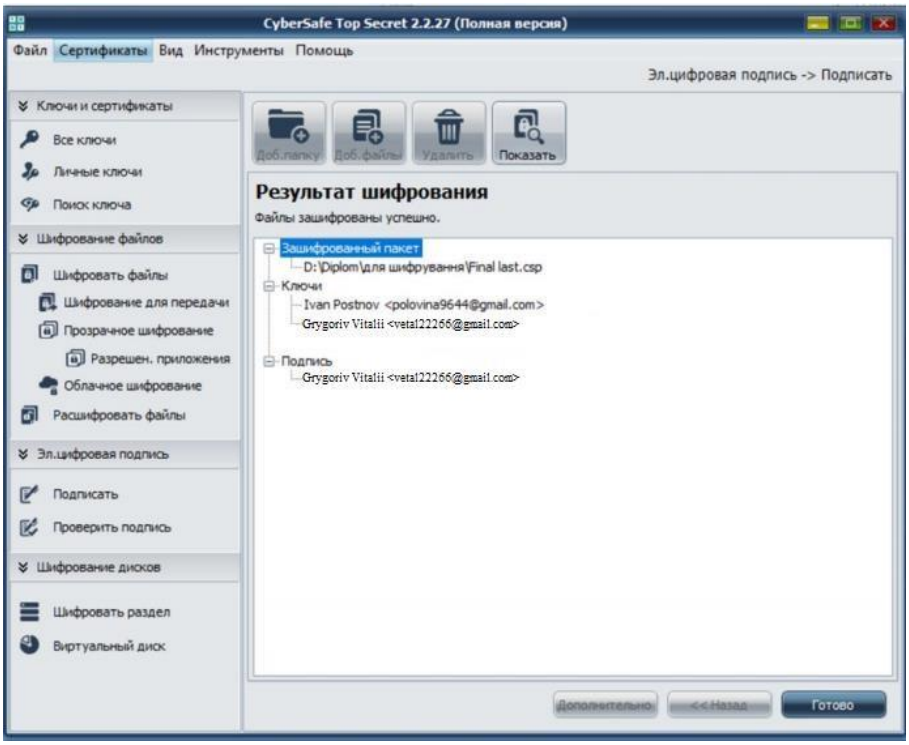

Рис. 3.22. Дані шифрування

Відеофайл, який потрібно зашифрувати, відкритий у текстовому редакторі, показано на рис. 3.23.

ў Ф±мZ‡,µ&Бs=¬Є 0 Cajwevл©c~ГГ¬рљЬ¬-Д?л]йЃкь3н®йзЗs~
‹€ ЉьС   J щр” W59Q— Ця4 {я'M.Sj¬€bNº ”!
ц¶k_э aj i ‡¦T¬' §Цt6Нў?Њ <W;A ёМУ^cBs9Pци-†79±35U>°
з нла † 3B+dн P дUYE = л…±P sE -LX,±}вЃ N p ”
µьc~O Й5Ъ X шX®?ќL O# §Ёaj:ћ rеўН ЫЙDќ8щю)@„

¬ nЭxpZjbVt8жъqў u ·4jкнњ› ›F5ЯЮJ FQsCђ¶33/qIи5—ЩГq0 Sл«ь
цвxx†Л ш€€; »r «e2й©Sѓ3©$ Pl«¤ '=ўЮJ A M BFµ9YUмµ€ c:*п>1
иqsїМ ~я EL+; РЌЛ°([I n·Км5ФЋh`™цZШ B‹ 6YA Цо $Et&®zrj`дCt}м/
#_np<sT¦Y KWG Гє 1ол$ ›^ў†Z?k= ¦ ¬
ЫЙ eГЊ0%=5 L·J ™lj€®) r„Q) Ра€Д#ТУЛфАш°
FJ' 0ђ‰FA
 o ЩЎ ®ф™ Щ{`3кы Ы-Ґінњ
'№{,;v†ўAt{Фњµ. ” p «Ћшsлґ5eY_}jяDs:Хм› Y+ ш
= Ёh¤nhk я>…Jќны :xSqTP°/w@Ц››ЁНа ^ogxэO 2
ЂЇ ™:sцвЊj4<ш IЦќ JстќХГЩIajйЋс ¦IыiECћ\ {jЭEtf§L}ДаZжqЎЃ·
~ Tj·Л@^—B8CIъсжТ‰…н L#ИUе*XUr»P°6\r niIZ¦jЊ†SrЛМЄЫРЎч?вЗ#Ш; !
&ЁЁЯп§w ¤ьPa¦jrђБ*
C5ky> Я¶ -ГutVH,}НхЛЩЦЦЋы4§-.тЃЮесэЦ2rё
> 5:СГ”Е®ПГМРw 7 РћШю~ jђ ё1,Зр ,KdJCQЊЊП\Э nub‹ # IXў
¶н Ё
эIў~ЌЋВ|k$}
GNº>ЖСS,“M^zlrBcrй M ьЙ вЩ)Пr1X,нЮ .Ц›§R‹ бЕЄ Ц‰ш}wwAF
M;тD ћ{ЃўЕцЃ.ДЄрлТХЩЖСWЯс@?Љ· CI, dхцЂN°
h В ”6°Ру®2@®39iЛBcMccґхЖБО>§5_P ?юZБЄf ндNBlIч“

Рис. 3.23. Відеофайл який потрібно зашифрувати

На рис. 3.24 вже можна переглянути відеофайл у зашифрованому вигляді з електронним цифровим підписом.

Рис. 3.24. Зашифрований відеофайл

Після шифрування розмір відеофайлу становив 32,27 МБ, а до шифрування - 32,37 МБ - документ було зменшено на 10 байт.

На рис. 3.24 представлено виділену текстову інформацію, яка свідчить про наявність локального ідентифікаційного ключа, за допомогою якого створено електронний цифровий підпис (ЕЦП).

3.5. Порівняння характеристик швидкості алгоритмів шифрування

Використовуючи графік, порівнюємо швидкості алгоритмів шифрування, які використовують різні методи та параметри шифрування. Для цього ми використовуватимемо відеофайл розміром 77,3 МБ і використовуємо різні типи шифрування:

- метод EPC;
- шифрування файлів паролем;
- метод шифрування файлів за допомогою відкритих ключів користувача.

Діаграми дозволяють порівнювати результати алгоритмів шифрування з урахуванням часу друг з одним. макс комп. - максимальне стиснення файлу, своєю чергою мін. - мінімальний стиск. Вертикальна вісь графіка – час шифрування файлу. Горизонтальний - алгоритми шифрування, що використовуються в експерименті з максимальною та мінімальною довжиною ключа.

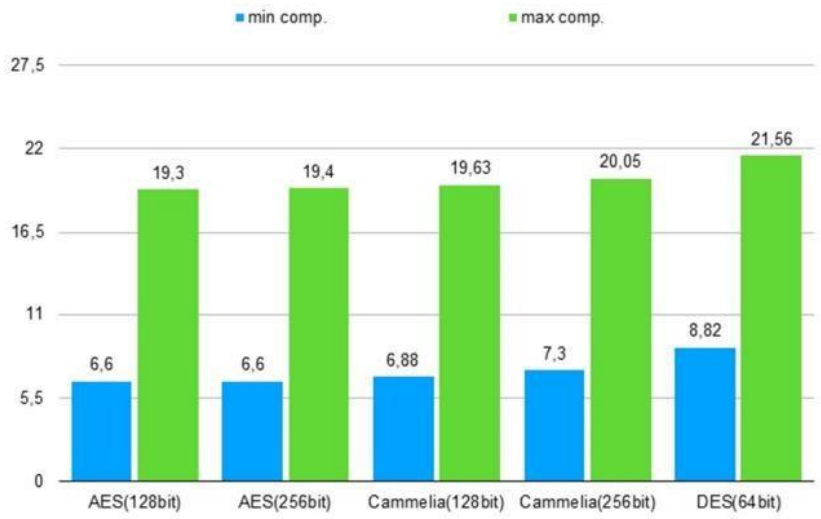

Рис. 3.25. Метод шифрування ЕЦП

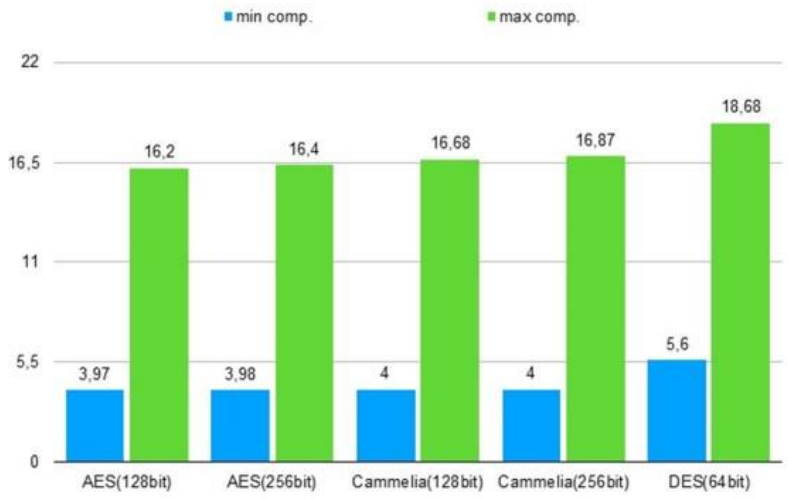

Рис. 3.26. Шифрування паролем

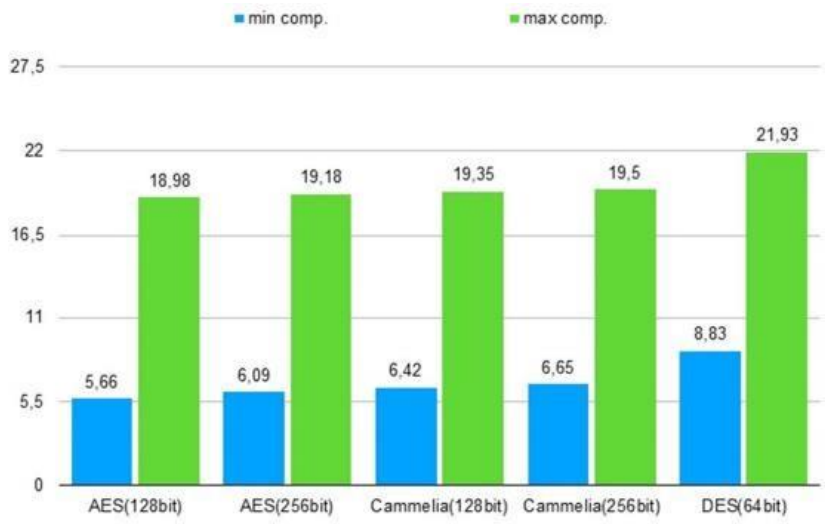

Рис. 3.27. Шифрування з відкритим ключем користувача

З діаграми видно, що метод шифрування AES з довжиною ключа 128 біт є найнадійнішим і найкращим алгоритмом серед показаних на рисунках 3.25 - 3.27.

3.6. Висновки

У цьому розділі наведено огляд того, як працює CyberSafe.

Після закінчення аналізу було обрано найнадійніший і найшвидший алгоритм — AES (128 біт).

РОЗДІЛ 4. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

Для перевірки ефективності шифрування було проведено дослідження з підрахунку пакетів даних перед і після шифрування.

Надалі ми будемо використовувати три способи надання інформації через системи зв'язку.

4.1. Захищена передача даних на основі VPN

Для передачі зашифрованого файлу ми використовуємо VPN для безпечної передачі даних. Для цього скористаємось програмою шифрування трафіку NordVPN.

NordVPN — це програма, призначена для захисту вашої інформації шляхом шифрування вашого з'єднання.

За допомогою цієї програми було змінено IP-адресу на рис. 4.1. Змінену адресу можна перевірити на https://2ip.ua.

Рис. 4.1. IP-адреса з використанням програми NordVPN

Для відстеження пакетної передачі зашифрованого файлу ми будемо використовувати PingPlotter - зручну і просту у використанні програму для Windows, мета якої - відстежити маршрут між вашою IP-адресою та будь-якою адресою. Цей продукт безперервно збиратиме і зберігатиме інформацію про загублені в дорозі посилки, після чого користувач зможе їх проаналізувати та вирішити ці проблеми.

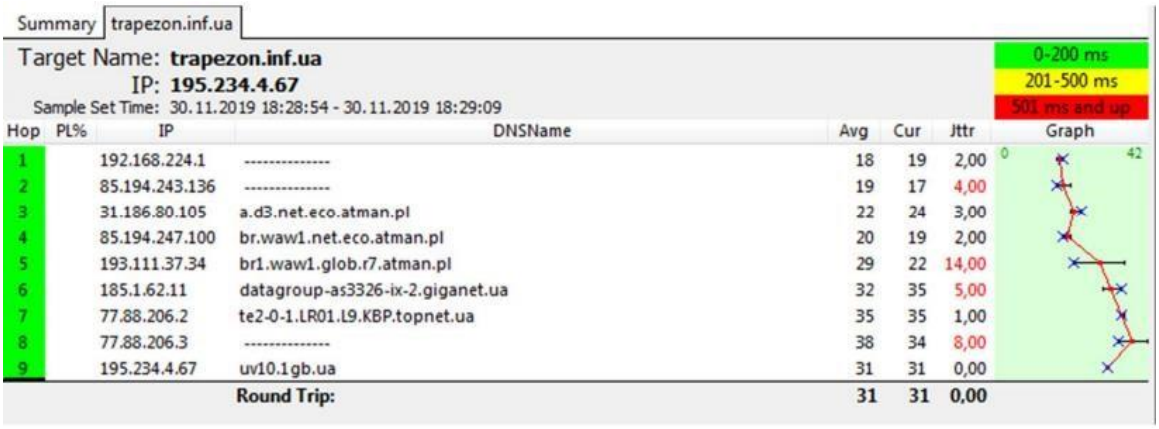

Рис. 4.2. Відстеження передачі даних

4.2. Передача даних на основі зміни IP

Щоб змінити IP-адресу мережі, з якої ми передаватимемо наші дані, ми будемо використовувати програму Tor. Цей продукт забезпечує анонімне з'єднання з функцією захисту від прослуховування.

Перший етап: заходимо до браузера Tor і після зміни IP-адреси створюємо нове посилання для передачі на сайт.

Другий етап: перевіряємо нову адресу за допомогою сайту https://2ip.ua.

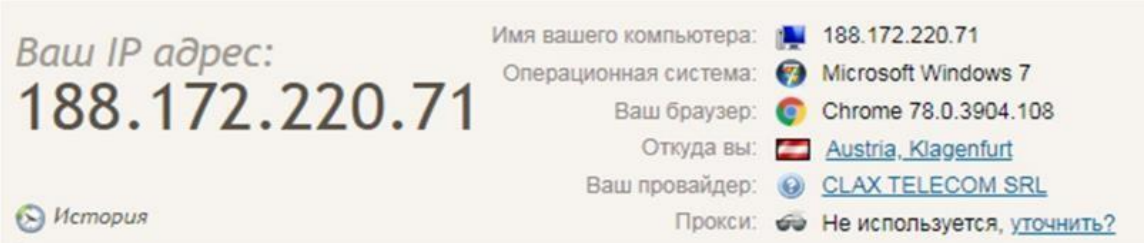

Рис. 4.3. Перевірка зміненого IP

Етап третій: після зміни IP-адреси входимо у програму WinSCP для подальшої передачі файлу на сервер.

Крок четвертий: слідуючи легенді відомої програми PingPlotter, простежимо шлях нашого завантаженого файлу по мережі. На графіці будуть показані точки, якими пройшов наш файл на рис. 4.4.

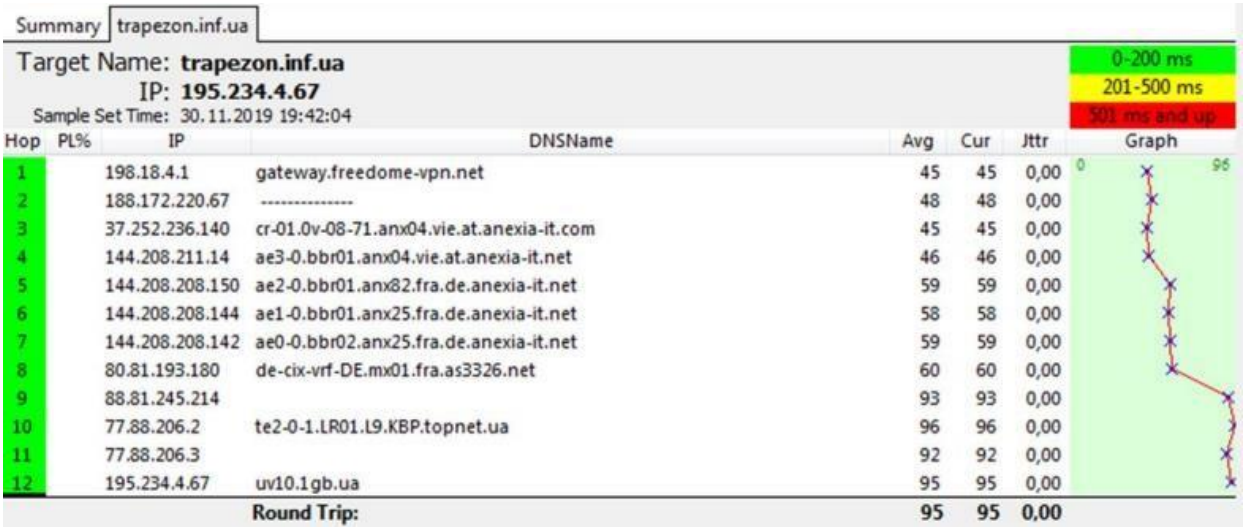

Рис. 4.4. Шлях переданого файлу по мережі

4.3. Незахищена передача даних (використовуючи відкритий http)

Ми будемо використовувати WinSCP для подальшого завантаження файлів. WinSCP – це безкоштовний клієнт SFTP, FTP та SCP з відкритим вихідним кодом. Основне завдання цієї програми – забезпечити безпечне копіювання файлів між потрібними серверами, здатними працювати з цими протоколами, та комп'ютером.

За допомогою цієї програми ми створимо підключення до сайту, представленого FTP-сервером, де є папка для завантаження файлів.

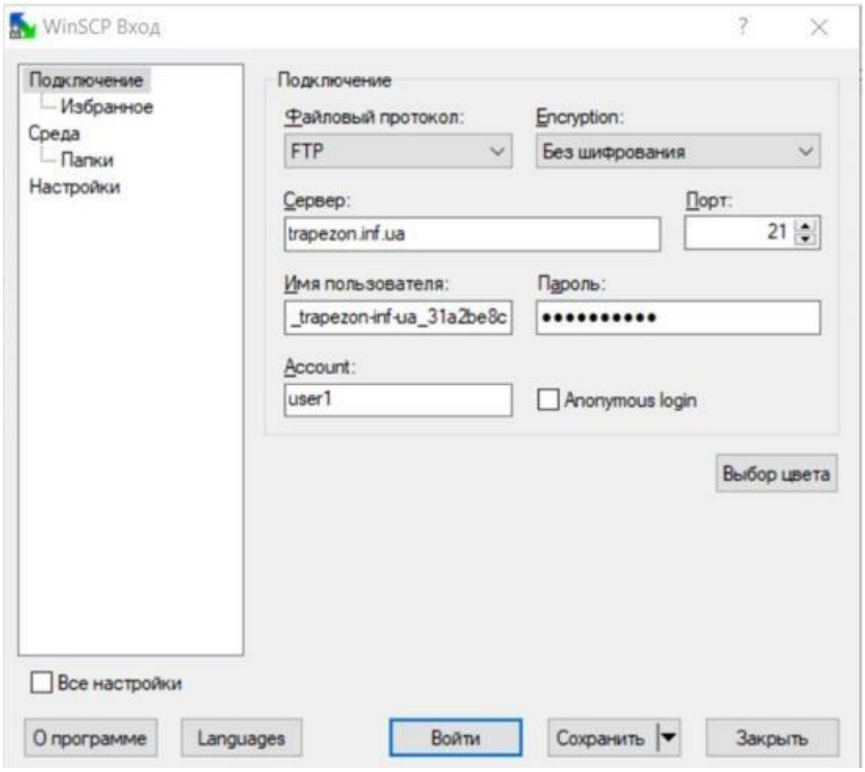

Рис. 4.5. Підключення до FTP-сервера

Після підключення до FTP-сервера знаходимо папку з файлами та потрібний нам зашифрований файл. Наші файли будуть передані їй.

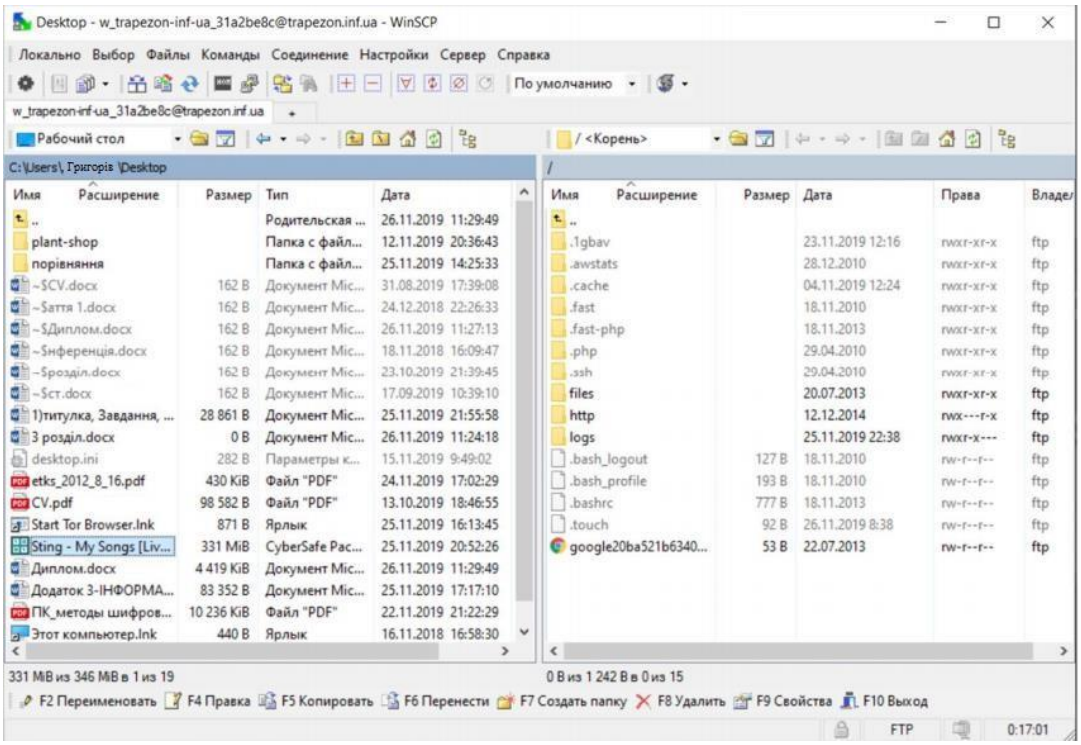

Рис. 4.6. Середовище FTP-сервера

Потім відкриваємо папку на сервері та переміщуємо туди наш зашифрований файл.

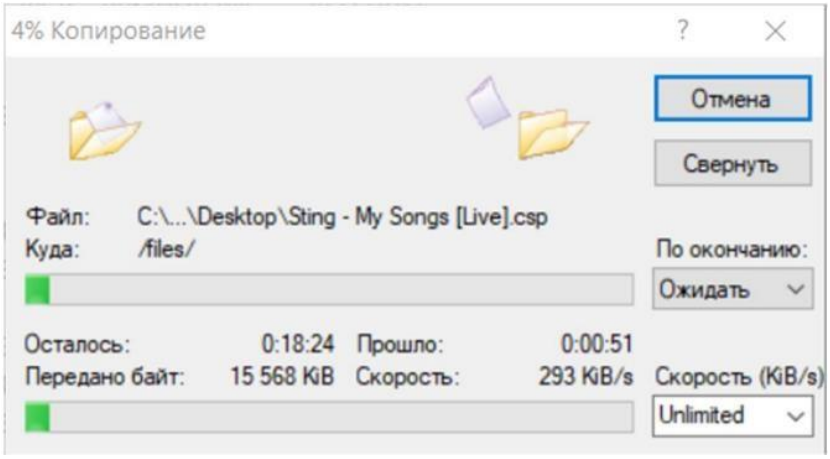

Рис. 4.7. Передача зашифрованого файлу

За допомогою програми PingPlotter проаналізуємо передачу файлових пакетів через мережу (рис. 4.8).

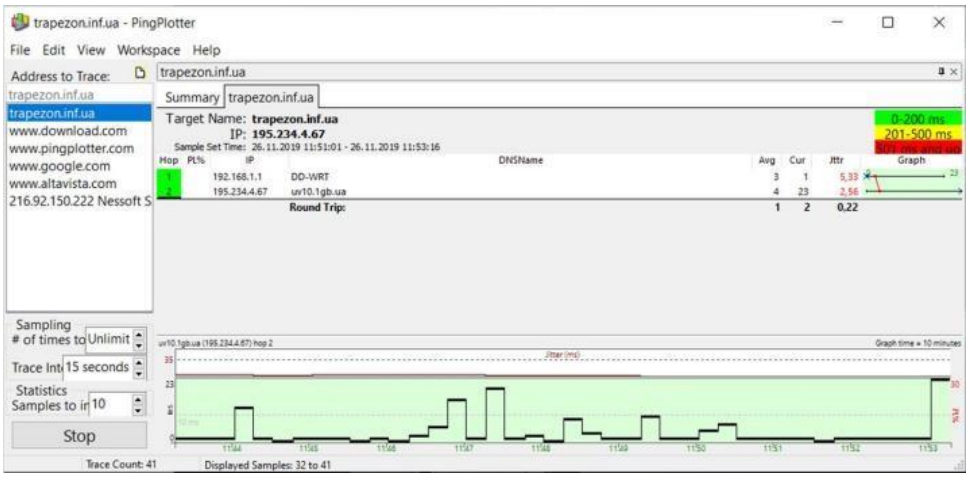

Рис. 4.8. Відстеження відправки

4.4. Порівняння кількості пакетів

Дані про відправлені пакети представлені в таблиці 4.1, а пакети після шифрування різноманітними алгоритмами, представлені в таблиці 4.2.

Таблиця 4.1

Кількість пакетів, що надіслані на шифрування

Тип даних	Кількість пакетів, які перенаправляються		
	Передача на основі VPN	Передача на основі IP (Tor)	Незахищена передача
Аудіо (.flac)	591987	667052	488100
Відео (.MOV)	195386	233623	159240
Зображення (jpg)	13763	14305	11162
Текстовий файл	7497	7867	6098

Таблиця 4.2

Кількість переданих пакетів даних після шифрування при різних алгоритмах

Тип даних	Кількість пакетів, які перенаправляються				
	Передача на основі VPN				
	AES 128bit	AES 256bit	Camellia 128bit	Camellia 256bit	DES 64bit
Аудіо	745231				
Відео	18473	18652	18745	18699	19124
Зображення	11169	11170	11166	11139	11146
	Передача на основі IP (Tor)				
Аудіо	752698				
Відео	247358	247567	247612	247589	2479458
Зображення	14151	14481	14689	18943	19916
	Незахищена передача				
Аудіо	488475	488586	489854	488863	488974
Відео	195874	196065	196258	195989	196265
Зображення	11169	11170	11166	11139	11146

4.5. Висновки

У цій частині роботи аналізується шифрування даних через мережу. При надсиланні інформації по мережі перевірялася зміна кількості пакетів даних після і до шифрування.

ВИСНОВКИ

1. Включено методи шифрування даних та їх характеристики.
2. Встановлено основні переваги щодо криптографії як засобу захисту інформації.
3. Проведено докладний аналіз передачі з використанням різних методів шифрування і передачі.
4. Подальший розвиток иагістерської роботи спрямований на удосконалення розробленого програмного забезпечення для задач стеганографії в аудіосигналах. Також планується розглянути це питання у відеофайлах.

СПИСОК ЛІТЕРАТУРИ

1. Гребенников В.В., История Криптологии & Секретной Связи [Электронный Ресурс] : глава 6.1, вступление в историю стеганографии, 2012. Режим доступа: http://cryptohistory.ru/book/chast-6-istoriyasteganografii/61-vstuplenie/
2. Забелин, М. А. Стегоанализ аудиоданных на основе методов сжатия // Вестник СибГУТИ. – 2010. – №1. – С. 41-46.
3. Зорин Е.Л., Чичиварин Н.В. Стеганография в САПР : учебное пособие. – Москва : МГТУ им. Н.Э. Баумана, 2013. – 90 с.
4. Импульсно-кодовая модуляция [Электронный ресурс]: Википедия. – 2016. – Режим доступа: https://ru.wikipedia.org/wiki/Импульснокодовая_модуляция
5. Кокорин П. П. О методах стегоанализа в аудиофайлах // Труды СПИИРАН. – 2007. – №4. – С. 239-246.
6. Очимов С. Ю. Стегоанализ аудиофайлов, базирующийся на алгоритмах сжатия // Вестник СибГУТИ. – 2010. – №1 . – С. 33-37.
7. Пакет статистических тестов NIST [Электронный ресурс]: Описание пакета и ссылка на скачивание. – 2014. – Гетейсберг. – Режим доступа: http://csrc.nist.gov/groups/ST/toolkit/rng/documentation_software.html
8. Структура wave файла [Электронный ресурс] : Audio Coding. – 2008. – Москва. – Режим доступа – http://audiocoding.ru/article/2008/05/22/wav-file-structure.html
9. DeepSound. [Электронный ресурс]: Описание ПО и ссылка на скачивание. – 2015. – Словения. – Режим доступа: http://jpinsoft.net/DeepSound/Overview.aspx
10. Digital Invisible Ink Toolkit. [Электронный ресурс] : Описание ПО и ссылка на скачивание. – 2008. – Новая Зеландия. – Режим доступа: https://sourceforge.net/projects/diit/?source=typ_redirect

11. Raggo M.T. Data Hiding: Exposing Concealed Data in Multimedia, Operating Systems, Mobile Devices and Network Protocols. M.T. Raggo. – Массачусетс : Syngress, 2012. – 270 с.
12. SilentEye. [Электронный ресурс] : Описание ПО и ссылка на
13. StegoStick beta. [Электронный ресурс]: Описание ПО и ссылка на скачивание. – 2013. – Сараево. – Режим доступа: https://sourceforge.net/projects/stegostick/
14. Technical Mujahid, a Training Manual for Jihadis. – 2010. – Нью-Йорк. – Режим доступа http://web.archive.org/web/20101013075447/www.jamestown.org/programs/gta/single/?tx_ttnews%5Btt_news%5D=1057&tx_ttnews%5BbackPid%5D=182&no_ca che=1
15. Wave file format – формат звукового файла wave [Электронный ресурс]: microsin. – 2011. – Москва. – Режим доступа: http://microsin.net/programming/pc/wav-format.html
16. Web Archive [Электронный ресурс]: Abdul Hameed Bakier, Xiao Steganography. [Электронный ресурс]: Описание ПО и ссылка на скачивание. – 2012. – Пекин. – Режим доступа – http://download.cnet.com/Xiao-Steganography/3000-2092_4-10541494.html

ДОДАТОК А. Вихідний код розробленого програмного забезпечення

```
unit WaveHideDiploma;
interface uses
  Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,
  Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.Buttons, Vcl.StdCtrls, Vcl.ComCtrls, zlib, ShellApi; type   bin = array[0..7] of 0..1;
  TypeSize = array[0..3] of byte;
  TForm5 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    BitBtn1: TBitBtn;
    OpenDialog1: TOpenDialog;
    Button3: TButton;
    SaveDialog1: TSaveDialog;
    Label1: TLabel;
    Memo1: TMemo;
    OpenDialog2: TOpenDialog;
    Button4: TButton;
    Button5: TButton;
    Button6: TButton;
    Edit1: TEdit;
    Edit2: TEdit;
    Label2: TLabel;
    Label3: TLabel;
    SaveDialog2: TSaveDialog;

    procedure Button1Click(Sender: TObject);     procedure Button3Click(Sender:
TObject);     Procedure HideWave(finn, foutn, fstn: string);     procedure
Button2Click(Sender: TObject);     procedure WaveUnHide(foutn, ffinn: string);
procedure BitBtn1Click(Sender: TObject);     procedure FormCreate(Sender:
TObject);     procedure Button4Click(Sender: TObject);     procedure
Button5Click(Sender: TObject);     procedure Button6Click(Sender: TObject);
private
    { Private declarations }    public
    { Public declarations }    end; var
  Form5: TForm5;
  Container, Ideal: string;    AnalyzeFileName: string;
implementation

{$R *.dfm}

function FromBytesToCardinal(by: TypeSize): cardinal; inline; begin   result := By[0] + (by[1] shl
8) + (by[2] shl 16) + (by[3] shl 24); end;

procedure FromCardinalToBytes(const AInput: cardinal; out by: TypeSize); inline; begin   by[0] := byte(AInput);   by[1] := byte(AInput shr 8);   by[2]
:= byte(AInput shr 16);   by[3] := byte(AInput shr 24); end;

function bytetobin(b: byte): bin; var   i: integer;
m: bin; begin   for i := 7 downto 0 do    begin
m[i]:= b mod 2;      b:= b div 2;    end;   result:= m;
end;

function bintobyte(b: bin): byte; var   ii, jj, pow:
integer; begin   for ii := 0 to 7 do     begin
pow:= 1;       for jj:= 1 to 7-ii do         pow:=
pow*2;       result:= pow * b[ii] + result;     end;
end;
//Реализация Алгоритма XOR

procedure first(fin, fout, fsteg: string); var   f_in: file of
byte;   f_out: file of byte;   f_steg: file of byte;   buf, i: byte;
d, df, ds: bin; begin   assign(f_in, fin);   assign(f_out, fout);
assign(f_steg, fsteg);

  reset(f_in);   rewrite(f_out);   reset(f_steg);



  while not eof(f_steg) do   begin


    if eof(f_in) then       reset(f_in);////

    blockread(f_in, buf, 1);
    d:= bytetobin(buf);     blockread(f_steg,
buf, 1);     df:= bytetobin(buf);     for i := 0
```

```
to 7 do       begin         if d[i] = df[i] then
ds[i]:= 1         else          ds[i]:= 0;       end;
buf:= bintobyte(ds) + 1;

   blockwrite(f_out, buf, 1);   end;


 closefile(f_in);   closefile(f_out);
closefile(f_steg);


end;

procedure last(fin, fout, fend: string); var   f_in: file of
byte;   f_out: file of byte;   f_end: file of byte;   buf, i:
byte;   d, df, ds: bin;
begin   assign(f_in, fin);
assign(f_out, fout);   assign(f_end,
fend);   reset(f_in);   reset(f_out);
rewrite(f_end);



 while not eof(f_out) do   begin     if
eof(f_in) then       reset(f_in);
blockread(f_in, buf, 1);     d:=
bytetobin(buf);     blockread(f_out, buf, 1);
df:= bytetobin(buf);

   for i := 0 to 7 do       begin         if df[i] = 1
then          ds[i]:= d[i];         if df[i] = 0 then
ds[i]:= 1 - d[i];       end;     buf:=
bintobyte(ds) + 1;     blockwrite(f_end, buf,
1);   end;


 closefile(f_in);   closefile(f_out);
closefile(f_end);



end;

procedure TForm5.WaveUnHide(foutn, ffinn: string); var
 F_out: file;   F_fin: file;   i, j, a, ia: integer;   d, df:
bin;   buf: byte;   col, consize: cardinal;   by:
typesize;   bps: integer; //Bytes per sample

begin   assignfile(F_out, foutn);
assignfile(F_fin, ffinn);   reset(F_out, 1);
rewrite(F_fin, 1);   consize:=
filesize(f_out);   for i := 0 to 33 do
begin
     Blockread(F_out, buf, 1);       dec(consize);
   end;

  blockread(F_out, buf, 1);    dec(consize);    BPS:=
buf div 8;    if foutn[length(foutn)] = 'p' then      bps:=
1;    memo1.lines.Add(inttostr(bps));

  for i := 35 to 104 - bps  do     begin
     Blockread(F_out, buf, 1);       dec(consize);
end;

 a:= 0;   i:= 0;

 while i < 4 do

   begin       j:= bps;                    //    Ділянка, що відповідає         while j > 1 do
//    за роботу с         begin                    //    різними          Blockread(F_out,
buf, 1);     //    bps wav          dec(consize);
         dec(j);                    //         end;

     blockread(f_out, buf, 1);
     dec(consize);       d:= bytetobin(buf);       df[a]:=
d[6];       inc(a);       df[a]:= d[7];       inc(a);       if a =
8 then         begin          a:= 0;          buf:=
bintobyte(df) + 1;//////          by[i]:= buf;          inc(i);
end;     end;
```

```
  col:= frombytestocardinal(by);   ia:= 0;

  while ia < col do     begin       if consize <= (bps*4) + 1 then       begin
memo1.Lines.Add('consize unhide =' + inttostr(consize));         closefile(f_out);
showmessage('Enter the next carrier');         if OpenDialog1.Execute then           foutn:=
OpenDialog1.FileName;         memo1.Lines.Add(foutn);         assignfile(f_out, foutn);
reset(f_out, 1);
       consize:= filesize(f_out);

       for i := 0 to 33 do          begin
          Blockread(F_out, buf, 1);
dec(consize);         end;

        blockread(F_out, buf, 1);          dec(consize);
BPS:= buf div 8;          if foutn[length(foutn)] = 'p' then
bps:= 1;          memo1.lines.Add(inttostr(bps));

      for i := 35 to 104 - bps  do        begin
        Blockread(F_out, buf, 1);
dec(consize);        end;
      end;


     a:=0;       while a < 8 do
begin
     ////////////////////////////////////       j:= bps;                        //  Ділянка відповідальна
while j > 1 do                //    за роботу з
       begin                        //    різними          Blockread(F_out, buf, 1);    //    bps wav
dec(consize);
         dec(j);                    //         end;
     ////////////////////////////////////


     blockread(f_out, buf, 1);
dec(consize);       d:= bytetobin(buf);
df[a]:= d[6];       inc(a);       df[a]:= d[7];
inc(a);

end;
         a:= 0;          buf:= bintobyte(df) + 1;
inc(ia);

         blockwrite(f_fin, buf, 1);   end;

  closefile(F_out);   closefile(F_fin);   last(Ideal,
ffinn, ffinn + 'end');   deletefile(ffinn);

end;

procedure TForm5.HideWave(finn, foutn, fstn: string); var
  f: textfile;   col, consize: cardinal;
d, df: bin;
  Buf, a: byte;   F_in: File;   buf1:
array[0..2047] of byte;
  F_out: file;   F_steg:
File;   i, j, it: integer;   by:
typesize;
  BPS: integer; //Bytes per Sample

begin   assignfile(F_in, finn);   assignfile(F_out, foutn);   first(Ideal,
'C:\Users\Public\mid.comp', fstn);   fstn:= 'C:\Users\Public\mid.comp';

  assignfile(F_steg, fstn);     ////fstn
  assignfile(f, 'c:\CourMyProj\tata.txt');   rewrite(f);
reset(F_in, 1);   reset(F_steg, 1);   rewrite(F_out, 1);   col:=
filesize(f_steg);   memo1.Lines.Add(inttostr(col));
FromCardinalToBytes(col, by);   consize:= filesize(f_in);
   for i := 0 to 33 do                ///     begin
      Blockread(F_in, buf, 1);       dec(consize);
/////DECCONSIZE       Blockwrite(F_out, buf, 1);     end;

   blockread(F_in, buf, 1);    dec(consize);
/////DECCONSIZE    BPS:= buf div 8;    if
finn[length(finn)] = 'p' then      bps:= 1;

   col:= filesize(f_steg);    memo1.lines.Add(inttostr(bps));
   Blockwrite(F_out, buf, 1);
```

```
   for i := 35 to 104 - bps  do     begin
     Blockread(F_in, buf, 1);       dec(consize);
/////DECCONSIZE       Blockwrite(F_out, buf, 1);     end;

  //В контейнер записується розмір файлу, що приховується

  for i := 0 to 3 do   begin
    d:= bytetobin(by[i]);     a:= 0;     while a < 8 do      begin        j:= bps;                    //
Ділянка відповідальна          while j > 1 do                  //     за роботу з          begin
//     різними             Blockread(F_in, buf, 1);      //     bps wav             dec(consize);
//
         Blockwrite(F_out, buf, 1);   //
         dec(j);                              //             end;
blockread(F_in,  buf,  1);                dec(consize);
df:=  bytetobin(buf);                    writeln(f,  buf);
df[6]:=  d[a];             inc(a);              df[7]:=  d[a];
inc(a);                              buf:=    bintobyte(df);
blockwrite(F_out, buf, 1);       end;   end;

  while col > 0 do

  begin     blockread(F_steg, buf, 1);     d:=
bytetobin(buf);
    a:= 0;     dec(col);

    /// Багатотомність

    if consize <= (bps*4) + 1 then     begin       memo1.Lines.Add('BPS= ' +
inttostr(BPS));       memo1.Lines.Add('consize = ' + inttostr(consize));

      while not eof(f_in) do       begin
blockread(f_in, buf, 1);         blockwrite(f_out,
buf, 1);       end;

      closefile(f_in);       closefile(f_out);

      showmessage('Not enough space. Choose next carrier.');       if opendialog1.Execute then
finn:= OpenDialog1.FileName;       assignfile(f_in, finn);       showmessage('Choose the name
of new encrypted file.');       if savedialog1.Execute then          foutn:= SaveDialog1.FileName;

      assignfile(f_out, foutn);       reset(f_in, 1);
rewrite(f_out, 1);       consize:= filesize(f_in);

      for i := 0 to 33 do       begin
      Blockread(F_in, buf, 1);       dec(consize);
Blockwrite(F_out, buf, 1);     end;

   blockread(F_in, buf, 1);     dec(consize);

   BPS:= buf div 8;     if finn[length(finn)] = 'p' then
bps:= 1;     memo1.lines.Add(inttostr(bps));
   Blockwrite(F_out, buf, 1);

   for i := 35 to 104 - bps  do     begin
      Blockread(F_in, buf, 1);       dec(consize);
Blockwrite(F_out, buf, 1);     end;

    end;

    while a < 8 do       begin         j:= bps;                    //    Ділянка відповідальна
while (j > 1)  do             //     за роботу з         begin                       //     різними
Blockread(F_in, buf, 1);     //     bps wav           dec(consize);               //
         Blockwrite(F_out, buf, 1);   //
         dec(j);                     //         end;

        blockread(F_in, buf, 1);         dec(consize);
df:= bytetobin(buf);         writeln(f, buf);
df[6]:= d[a];         inc(a);         df[7]:= d[a];
inc(a);         buf:= bintobyte(df);
blockwrite(F_out, buf, 1);       end;   end;

  it:= (filesize(f_in) - filepos(f_in)) div 2048;   for i := 1 to it do
begin       blockread(f_in, buf1, 2048);       blockwrite(f_out, buf1,
2048);     end;

  while not eof(f_in) do     begin
blockread(f_in, buf, 1);       blockwrite(f_out,
buf, 1);     end;
```

```
  closefile(F_in);   closefile(F_out);   closefile(F_steg);
showmessage('File is successfully hidden.'); end;

//UnHide procedure TForm5.Button2Click(Sender: TObject); var   f1, f2: string;
begin     if OpenDialog1.Execute     then        f1:= OpenDialog1.FileName;     if
SaveDialog2.Execute then     f2:= SaveDialog2.FileName;   WaveUnHide(f1, f2);
end;

//Choose a carrier procedure TForm5.Button3Click(Sender: TObject); var   f:
file of byte;   i: integer;   bps: integer;   buf: byte;

begin   if OpenDialog1.Execute   then       container:=
OpenDialog1.FileName;   assignfile(f, container);   reset(f);
   for i := 0 to 33 do                ///     begin
      Blockread(f, buf, 1);     end;

   blockread(f, buf, 1);    BPS:= buf div 8;
  memo1.Lines.Add('Container size: ' + inttostr((-100 - 4*bps + (filesize(f) div (bps*4))) div
1024) + 'KB');   closefile(f);
end;


///////СТЕГОАНАЛІЗ!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!//////////////////////////////////

procedure TForm5.Button4Click(Sender: TObject);
  var     ftex: string;

function empty(fin: string): double; var   f_n: file of
byte;   col, cole: int64;   buf, i: byte;   d, d1, d2, b, b1,
b2: bin;
  bps, j: integer; begin   col:= 0;   cole:= 0;
assignfile(f_n, fin);   reset(f_n);   for i := 0 to 33
do               ///     begin       Blockread(F_n, buf,
1);     end;

   blockread(F_n, buf, 1);
   BPS:= buf div 8;

   for i := 35 to 102 - bps  do               ///     begin
Blockread(F_n, buf, 1);     end;

  while not eof(f_n) do   begin
Blockread(F_n, buf, 1);     if buf = 0 then
inc(cole);     inc(col);   end;

  result:= cole / col;
closefile(f_n); end;

procedure sumoflsb(fin: string; fout: string); var   f_in: file of byte;
f_out: textfile;   buf, i: byte;   d, df: bin;   col, bps, j: integer; begin
col:= 0;   assignfile(f_in, fin);   reset(f_in);

  assignfile(f_out, fout);   rewrite(f_out);
   for i := 0 to 33 do                ///     begin
      Blockread(F_in, buf, 1);     end;

   blockread(F_in, buf, 1);
   BPS:= buf div 8;

   for i := 35 to 104 - bps  do               ///     begin
      Blockread(F_in, buf, 1);     end;

  while not eof(f_in) do
  begin     j:= bps;                    //    Ділянка відповідальна        while not eof(f_in) and (j > 1) do
//    за роботу з        begin                         //    різними          Blockread(F_in, buf, 1);     //    bps
wav
         dec(j);                    //        end;

      if eof(f_in) then         exit;

      blockread(f_in, buf, 1);       d:=
bytetobin(buf);       write(f_out, d[6]);
write(f_out, d[7]);   end;   closefile(f_in);
closefile(f_out); end;

begin

  //ftex:= 'C:\ Текстові документи та все, що з ними пов'язано \ИМ\!!Магістарська робота\Stegora Wave
```

```
Hide\results\res.txt';   if opendialog1.Execute
then
       AnalyzeFileName := OpenDialog1.FileName;   if savedialog2.Execute
then     ftex:= savedialog2.FileName;

  memo1.Lines.Add(AnalyzeFileName);   memo1.Lines.Add(floattostr(empty(AnalyzeFileName)));   sumoflsb(AnalyzeFileName, ftex);
  shellexecute(Handle, 'open', 'C:\ Текстові документи та все, що з ними пов'язано \ИМ\!!Магістерська робота
АСУ 2021\!!Стегоаналіз\NIST-statistical-test-suite-UI2.1.2\Release\NIST_UI.exe', nil, nil,
SW_SHOWNORMAL); end;

procedure TForm5.Button5Click(Sender: TObject);
  var     f1: textfile;
f2: textfile;     c:
double; begin
   assignfile(f2, 'C:\ Текстові документи та все, що з ними пов'язано \ИМ\!! Магістарська робота АСУ 2021\Stegora Wave
Hide\results\results.txt');
   assignfile(f1, 'C:\ Текстові документи та все, що з ними пов'язано \ИМ\!! Магістарська робота \Stegora Wave
Hide\Win32\Debug\experiments\AlgorithmTesting\ApproximateEntropy\results.txt');
   reset(f1);   read(f1, c);
closefile(f1);

   rewrite(f2);          writeln(f2,   AnalyzeFileName);
writeln(f2,'Approx: ',  c);

   assignfile(f1, 'C:\ Текстові документи та все, що з ними пов'язано \ИМ\!! Магістарська робота \Stegora Wave
Hide\Win32\Debug\experiments\AlgorithmTesting\BlockFrequency\results.txt');
   reset(f1);    read(f1, c);     closefile(f1);
writeln(f2,'Block: ',   c);     assignfile(f1,
'C:\ Текстові документи та все, що з
ними пов'язано \ИМ\!! Магістарська
робота             \Stegora            Wave
Hide\Win32\Debug\experiments\Algorit
hmTesting\CumulativeSums\results.txt')
;
   reset(f1);               read(f1,    c);               closefile(f1);
writeln(f2,'CumulativeSums: ',  c);


   assignfile(f1, 'C:\ Текстові документи та все, що з ними пов'язано \ИМ\!! Магістарська робота \Stegora Wave
Hide\Win32\Debug\experiments\AlgorithmTesting\FFT\results.txt');
   reset(f1);                    read(f1,     c);
closefile(f1);    writeln(f2,'FFT: ',  c);

   assignfile(f1, 'C:\ Текстові документи та все, що з ними пов'язано \ИМ\!! Магістарська робота АСУ 2021\Stegora
Wave Hide\Win32\Debug\experiments\AlgorithmTesting\Frequency\results.txt');
   reset(f1);         read(f1,  c);         closefile(f1);
writeln(f2,'Frequency: ',  c);

   assignfile(f1, 'C:\ Текстові документи та все, що з ними пов'язано \ИМ\!! Магістарська робота АСУ 2021\Stegora
Wave Hide\Win32\Debug\experiments\AlgorithmTesting\LongestRun\results.txt');
   reset(f1);          read(f1,   c);          closefile(f1);
writeln(f2,'LongestRun: ',  c);

   assignfile(f1, 'C:\ Текстові документи та все, що з ними пов'язано \ИМ\!! Магістарська робота АСУ 2021\Stegora
Wave Hide\Win32\Debug\experiments\AlgorithmTesting\Rank\results.txt');
   reset(f1);   read(f1, c);   closefile(f1);
writeln(f2,'Rank: ',  c);

   assignfile(f1, 'C:\Текстовые документы и все, что с ними связано\ИМ\!! Магістарська робота АСУ 2021\Stegora
Wave Hide\Win32\Debug\experiments\AlgorithmTesting\Runs\results.txt');
   reset(f1);   read(f1, c);   closefile(f1);
writeln(f2,'Runs: ',  c);

   assignfile(f1, 'C:\ Текстові документи та все, що з ними пов'язано \ИМ\!! Магістарська робота АСУ 2021\Stegora
Wave Hide\Win32\Debug\experiments\AlgorithmTesting\Serial\results.txt');
   reset(f1);    read(f1, c);    closefile(f1);
writeln(f2,'Serial: ',  c);    closefile(f2);


////////////////


   shellexecute(Handle, 'open', 'C:\windows\notepad.exe', 'C:\ Текстові документи та все, що з ними пов'язано \ИМ\!! Магістарська робота
\Stegora Wave Hide\results\results.txt', nil,
SW_SHOWNORMAL);

end;

//////!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!/////////////
```

```
procedure TForm5.Button6Click(Sender: TObject); var   fn: string;   f, f1, f2: file
of byte;   k, r, ii, iii, j: integer;   b0:  double;   s: byte;   fx, fy: array of double;
reresult: integer;   sig: double;

  procedure CompressFile(const Source, Dest : String); var
   SourceFile, DestFile : TFileStream;    compr :
TCompressionStream;    bytecount : Integer; begin
   SourceFile := TFileStream.Create(Source, fmOpenRead);    DestFile :=
TFileStream.Create(Dest,fmCreate);
   try
     bytecount := SourceFile.Size;
     //store the original size in bytes
     DestFile.Write(bytecount, SizeOf(bytecount));
     //create the compression stream (after storing the bytecount!)      compr :=
TCompressionStream.Create(clMax, DestFile);
     try
       //compress the content of the file        compr.CopyFrom(SourceFile,bytecount);
     finally        compr.Free;      end;
finally
     SourceFile.Free;
DestFile.Free;    end; end;

  procedure HideWaveRandom(finn, foutn: string); var
  f: textfile;   col, consize: cardinal;
d, df: bin;
  Buf, st, a: byte;   F_in: File;   buf1:
array[0..2047] of byte;   stop: array[0..3] of
byte;
  F_out: file;
   i, j, it: integer;   by:
typesize;   mid: string;
  BPS: integer; //Bits per Sample

  count, current: integer;

begin   assignfile(F_in, finn);
assignfile(F_out, foutn);

  randomize;

  assignfile(f, 'c:\CourMyProj\tata.txt');   rewrite(f);

  reset(F_in, 1);

  rewrite(F_out, 1);



  count:= col;

  //memo1.Lines.Add(inttostr(col));
  FromCardinalToBytes(col, by);
   for i := 0 to 33 do                ///     begin
      Blockread(F_in, buf, 1);       Blockwrite(F_out, buf,
1);     end;

   blockread(F_in, buf, 1);

   BPS:= buf div 8;    if finn[length(finn)] = 'p'
then      bps:= 1;

   Blockwrite(F_out, buf, 1);    for i := 35 to 104 - bps  do
///     begin
      Blockread(F_in, buf, 1);       Blockwrite(F_out, buf,
1);     end;

  //memo1.Lines.Add(inttostr(col) + '   = col    ' + inttostr(filesize(f_in)));   memo1.lines.add('bps = ' +
inttostr(bps));   col:= -132 + (filesize(f_in) div (bps));
  while col > 0 do

  //while not eof(f_in) do


  begin     dec(col);

      begin         j:= bps;                    //    Ділянка відповідальна         while j > 1 do
//    за роботу з         begin
          //if not eof(f_in) then                          //    різними
```

```
         Blockread(F_in, buf, 1);          //    bps wav           Blockwrite(F_out, buf, 1);   //
         dec(j);                      //         end;
       //if not eof(f_in) then

       blockread(F_in, buf, 1);         df:=
bytetobin(buf);          writeln(f, buf);/////////

       //////////
       df[6]:= random(2);
        df[7]:= random(2);          ////////////////////////////////////

       buf:= bintobyte(df);         blockwrite(F_out,
buf, 1);       end;   end;

  while not eof(f_in) do     begin       blockread(f_in,
buf, 1);      // memo1.Lines.Add('ola la');
blockwrite(f_out, buf, 1);     end;

  closefile(F_in);
closefile(F_out);
closefile(f); end;

begin

 begin   if opendialog2.Execute then     fn:= opendialog2.FileName;   memo1.Lines.Add(fn);
  assignfile(f, fn);   reset(f);   r:= filesize(f) div strtoint(edit2.Text);   for ii := 1 to strtoint(Edit2.Text)-1 do     begin
assignfile(f1, 'c:\decrypt\slices\slice' + inttostr(ii) + '.wav');       rewrite(f1);       for j := 0 to r do         begin           read(f, s);
write(f1, s);         end;       closefile(f1);     end;   assignfile(f1, 'c:\decrypt\slices\slice' + Edit2.Text + '.wav');   rewrite(f1);   while
not eof(f) do   begin     read(f,s);     write(f1,s);   end;   closefile(f1);   setlength(fx, 1);   for ii := 1 to strtoint(Edit2.Text) do
begin       compressfile('c:\decrypt\slices\slice' + inttostr(ii) + '.wav', 'c:\decrypt\comp.zip');       assignfile(f1,
'c:\decrypt\comp.zip');       reset(f1);
      assignfile(f2, 'c:\decrypt\slices\slice' + inttostr(ii) + '.wav');       reset(f2);       setlength(fx,
length(fx) + 1);       memo1.Lines.Add(inttostr(filesize(f1)));
memo1.Lines.Add(inttostr(filesize(f2)));

      fx[ii]:= (filesize(f1) / filesize(f2));       closefile(f1);
closefile(f2);       deletefile('c:\decrypt\comp.zip');     end;
closefile(f);
   ////////////// ПОВТОРЕННЯ ОПЕРАЦІЇ ДЛЯ ПОВНІСТТЮ ЗАПОВНЕНОГО
ФАЙЛУ!!!!!!/////////////

  hidewaverandom(fn, 'c:\decrypt\me.wav');

  fn:= 'c:\decrypt\me.wav';


  assignfile(f, fn);   reset(f);   r:= filesize(f) div strtoint(edit2.Text);    for ii := 1 to strtoint(Edit2.Text)-1
do     begin       assignfile(f1, 'c:\decrypt\slices\filledslice' + inttostr(ii) + '.wav');       rewrite(f1);       for j
:= 0 to r do         begin           read(f, s);           write(f1, s);
       end;       closefile(f1);     end;

  assignfile(f1, 'c:\decrypt\slices\filledslice' + Edit2.Text + '.wav');   rewrite(f1);   while not eof(f) do   begin     read(f,s);     write(f1,s);
end;   closefile(f1);   setlength(fy, 1);   for ii := 1 to strtoint(Edit2.Text) do     begin       compressfile('c:\decrypt\slices\filledslice' +
inttostr(ii) + '.wav', 'c:\decrypt\comp.zip');       assignfile(f1, 'c:\decrypt\comp.zip');       reset(f1);       assignfile(f2,
'c:\decrypt\slices\filledslice' + inttostr(ii) + '.wav');       reset(f2);       setlength(fy, length(fy) + 1);       fy[ii]:= (filesize(f1) / filesize(f2));
closefile(f1);       closefile(f2);       deletefile('c:\decrypt\comp.zip');     end;     closefile(f);   memo1.Lines.Add(floattostr(sig));
  reresult:= 0;   for ii := 1 to strtoint(Edit2.Text) do     begin       sig:= abs(fx[ii] - fy[ii]);
memo1.Lines.Add(inttostr(ii) + ' f(x) = ' + floattostr(fx[ii]));       memo1.Lines.Add(inttostr(ii) + '
f(y) = ' + floattostr(fy[ii]));       memo1.Lines.Add('delta = ' + floattostr(sig));

      if sig < strtofloat(Edit1.Text) then       begin           reresult:= reresult + 1;           memo1.Lines.Add('|f(X, ' + inttostr(ii) + ') -
f(Y, ' + inttostr(ii) + ') < Sigma' );         end       else         begin           reresult:= reresult - 1;           memo1.Lines.Add('|f(X, ' +
inttostr(ii) + ') - f(Y, ' + inttostr(ii) + ') > Sigma' );         end;     end;

end;

end;

////////////КІНЕЦЬ!!!!!!!!!!!!!!!!!!!!!!!!!!!!!///////////////

////////Кінец Стегоаналізу

procedure TForm5.FormCreate(Sender: TObject);

begin

end;
```

```
//Keyfile procedure TForm5.BitBtn1Click(Sender: TObject); begin   if
OpenDialog1.Execute then     Ideal:= OpenDialog1.FileName; end;
//WaveHide procedure TForm5.Button1Click(Sender: TObject); var   steg, fin:
string; begin   if OpenDialog1.Execute then     steg:= OpenDialog1.FileName;

  if SaveDialog1.Execute then     fin:=
SaveDialog1.FileName;   HideWave(Container, Fin,
Steg); end; end.
```