Міністерство освіти і науки України
Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій
Кафедра кібербезпеки та захисту інформації

ДОПУСТИТИ ДО ЗАХИСТУ:
В.о. завідувач кафедри
кібербезпеки та захисту інформації
_____ Іван ПАРХОМЕНКО
“_____” червня 2023 р.

ПОЯСНЮВАЛЬНА ЗАПИСКА

кваліфікаційної роботи

бакалавра
(назва освітнього ступеня)

галузь знань 12 Інформаційні технології
(шифр і назва галузі знань)

спеціальність 125 «Кібербезпека»
(код і назва спеціальності)

освітній ступень бакалавр

освітня програма Кібербезпека
(назва освітньої програми)

на тему: Засоби обробки метаданих для захищеної передачі мультимедіа трафіку

Виконавець: студент 4 курсу, групи КБ-42

_____ **Віталій КОСТЮЧЕНКО**
(підпис) (ім’я прізвище)

	Прізвище, ініціали	**Підпис**
Керівник роботи	Іван ПАРХОМЕНКО	

Нормоконтроль	Лариса МИРУТЕНКО	

Київ 2023

Міністерство освіти і науки України
Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій
Кафедра кібербезпеки та захисту інформації

ЗАТВЕРДЖЕНО:
В.о. завідувач кафедри
кібербезпеки та захисту інформації
_____ Сергій ТОЛЮПА
«21» листопада 2022 р.

ЗАВДАННЯ
на виконання кваліфікаційної роботи

спеціальності 125 Кібербезпека
(код і назва спеціальності)

освітньої програми Кібербезпека
(назва освітньої програми)

Студенту **КБ-42** (група) **Костюченку Віталію Сергійовичу** (прізвище ім'я по-батькові)

Тема кваліфікаційної роботи Засоби обробки метаданих для захищеної передачі мультимедіа трафіку

1. ПІДСТАВИ ДЛЯ ПРОВЕДЕННЯ РОБОТИ

Тема кваліфікаційної роботи затверджена на засіданні кафедри кібербезпеки та захисту інформації протокол №3 від 20.10.2022 р.

2. ВИХІДНІ ДАНІ ДЛЯ ПРОВЕДЕННЯ РОБІТ

Класифікація метаданих, стандарти та формати метаданих для мультимедійних файлів, процес забезпечення доступності, цілісності, конфіденційності мультимедійних файлів

3. ЗМІСТ РОЗРАХУНКОВО-ПОЯСНЮВАЛЬНОЇ ЗАПИСКИ

Нормативно-правова база у сфері захисту інформації, класифікація метаданих, стандарти та формати метаданих для мультимедійних файлів, загрози конфіденційності, пов'язані з метаданими, методи та техніки обробки метаданих, архітектура програмного модуля, програмний застосунок

4. ВИМОГИ ДО РЕЗУЛЬТАТІВ ВИКОНАННЯ РОБОТИ

Практична цінність Розроблений програмний застосунок, який забезпечує доступність, цілісність та конфіденційність метаданих у мультимедійних файлах шляхом їх обробки, та передачі по зашифрованим каналам зв'язку

5. ДАТА ВИДАЧІ ЗАВДАННЯ

Дата видачі завдання: 24 жовтня 2022 року

Завдання видав _____ (підпис) Іван ПАРХОМЕНКО (ім'я, прізвище)

Завдання прийняв до виконання _____ (підпис) Віталій Костюченко (ім'я, прізвище)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Найменування етапів робіт	Строки виконання робіт (початок-кінець)	Відмітка про виконання
1	Уточнення постановки задачі	21.11.2022 – 28.01.2023	*виконано*
2	Аналіз відкритих джерел	29.01.2023 – 11.02.2023	*виконано*
3	Обґрунтування вибору рішення	12.02.2023 – 15.02.2023	*виконано*
4	Поняття метаданих та їх класифікація	16.02.2023 – 04.03.2023	*виконано*
5	Аналіз методів захисту конфіденційності метаданих мультимедійних файлів	05.03.2023 – 21.03.2023	*виконано*
6	Визначення недоліків та проблем захисту конфіденційності при передачі мультимедійних ресурсів в мережі Інтернет	22.03.2023 – 08.04.2023	*виконано*
7	Вироблення програмного застосунку для підвищення захищеності метаданих мультимедійних файлів	09.04.2023 – 10.05.2023	*виконано*
8	Оформлення пояснювальної записки	11.05.2023 – 27.05.2023	*виконано*
9	Підготовка до захисту кваліфікаційної роботи	28.05.2023 – 12.06.2023	*виконано*

Завдання видав _____ (підпис) Іван ПАРХОМЕНКО (ім'я, прізвище)

Завдання прийняв до виконання _____ (підпис) Віталій Костюченко (ім'я, прізвище)

Термін подання кваліфікаційної роботи до ЕК 12 червня 2023 року

РЕФЕРАТ

Пояснювальна записка дипломної роботи складається зі вступу, 3 розділів, загальних висновків, списку використаних джерел та додатків. Основний текст займає 61 сторінку, включає в себе зміст, вступ, 3 розділи дипломної роботи, висновки та список джерел. Крім того, робота містить 3 додатки із загальною кількістю сторінок 13. У пояснювальній записці дипломної роботи міститься 2 рисунки та 32 літературних джерела.

Метою роботи є розробка програмних засобів, які підвищує доступність, цілістність та конфіденційність при роботі з метаданими та передачі пов’язаними з ними мультимедійними файлами.

Об’єктом дослідження є процес реалізації захищеної передачі метаданих мультимедія трафіку, можливостей підвищення доступності як перегляду метаданих мультимедійних файлів, та забезпечення конфіденційності у метаданих для передачі мультимедія трафіку.

Предметом дослідження є набір механізмів, що реалізують процес захисту основних аспектів інформації метаданих мультимедійних файлів.

Методи дослідження: аналіз відкритих джерел; аналіз існуючих методів та технік обробки метаданих; аналіз механізмів захищеної передачі мультимедійного трафіку.

Практичною цінністю є розроблений програмний модуль для перегляду та обробки метаданих для передачі пов’язаних з ними мультимедійних файлів.

Ключові слова: метадані, захищена передача, кібербезпека, програмний застосунок, класифікація метаданих, стандарти та формати метаданих, захист персональних даних, загрози конфіденційності, C, Python, EXIF.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕННЬ

ПЗ	–	програмне забезпечення
JPEG	–	Joint Photographic Experts Group
TIFF	–	Tagged Image File Format
PSD	–	Photoshop Document
EXIF	–	Exchangeable Image File Format
GDPR	–	General Data Protection Regulation
OSINT	–	Open source intelligence
IPTC	–	International Press Telecommunications Council
XMP	–	Extensible Metadata Platform
ASCII	–	American standard code for information interchange
FTP	–	File Transfer Protocol
TCP	–	Transmission Control Protocol
IP	–	Internet Protocol
HTTP	–	Hyper Text Transfer Protocol
URI	–	Uniform Resource Identifier
IMAP	–	Internet Message Access Protocol
POP3	–	Post Office Protocol 3
MIME	–	Multipurpose Internet Mail Extensions
SMTP	–	Simple Mail Transfer Protocol
TLS	–	Transport Layer Security
RFC	–	Request for Comments
SSL	–	Secure Sockets Layer
SASL	–	Simple Authentication and Security Layer

ВСТУП

У сучасному світі швидкого розвитку інформаційних технологій, передача мультимедійного трафіку займає центральне місце в багатьох аспектах нашого життя. Завдяки безперервному зростанню швидкостей передачі даних та покращенню технологій мережі Інтернет, ми маємо можливість обмінюватись великими обсягами мультимедійної інформації, такої як відео, аудіо та зображення, у режимі реального часу. Згідно з даними Rise Above Research, у 2021 році в усьому світі було зроблено 1,2 трильйона фотографій, а в 2022 році – 1,72 трильйона. У 2023 році це число зросте до 1,81 трильйона. До 2025 року щороку буде зроблено понад 2 трильйони фотографій [1].

Зростання обсягів мультимедійного трафіку також поставило перед нами нові виклики і проблеми. Однією з таких проблем є збереження конфіденційності особистих даних під час передачі мультимедійних файлів. Метадані, які супроводжують ці файли, містять інформацію про їх вміст, формат, авторство та інші деталі. Однак, нерідко вони також можуть містити компрометуючу інформацію, яка може бути використана для розкриття особистого життя та порушення приватності користувачів. Незважаючи на те, що створення й обробка певної кількості метаданих може бути виправданим і фактично принести користь кінцевому користувачеві, деякі метадані можуть стати занадто особистими. Наприклад, фотографії з сімейного відпочинку можуть передавати географічні координати, а також інформацію про використовуваний телефон або камеру. (Таким чином сталкери можуть визначити своїх жертв навіть на великій відстані.) Крім того, метадані можуть включати й інших людей. Наприклад, метадані програми можуть містити інформацію про пристрої поблизу, автоматично отриману нашими смартфонами. У друга є наручний годинник із Bluetooth? Програма на вашому телефоні могла помітити та зареєструвати це також. Якщо ми можемо належним чином контролювати та керувати метаданими, ми зможемо користуватися швидшою, стабільнішою та персоналізованою технологією. Однак якщо дані неконтрольовано збираються та передаються третім сторонам без

чіткого відома та згоди, це може становити серйозний ризик для конфіденційності та керування ідентифікацією.

Згідно з дослідженням IPSOS проведеним у 2019 році - лише кожен третій дорослий у всьому світі має уявлення про те, скільки персональних даних компанії зберігають про нього (35%) або що вони з ними роблять (32%). У Великій Британії 30% опитаних стверджують, що знають багато про те, якими даними про них володіють компанії, і лише 27% впевнені в тому, що вони роблять з цими даними: значно нижче середніх світових показників. Тим не менш, британці досягли кращих результатів, ніж громадяни з інших розвинутих економік, таких як Канада, Німеччина та Японія, де в середньому 20% кажуть, що вони добре уявляють, що компанії роблять з їх особистою інформацією [2]. Компанії, що займаються передачею мультимедійного трафіку, повинні вживати заходів для забезпечення конфіденційності та захисту особистих даних своїх користувачів. Відсутність таких заходів може мати серйозні наслідки, включаючи порушення законодавства щодо захисту персональних даних та втрату довіри користувачів [3]. Користувачі, мають право знати, яка інформація про них збирається, як вона використовується і як вони можуть контролювати цей процес.

Особливо важливою проблемою метаданих є їх актуальність у сучасних військових конфліктах, зокрема в контексті війни в Україні. Інформація, яка стає відомою, може мати вирішальне значення для військових операцій і наслідки її неналежного використання можуть бути критичними як для інформаційної безпеки, так і для загальної безпеки. Особливу увагу слід приділити інформації, яка дозволяє визначити місцезнаходження людей або об'єктів, наприклад, для коригування вогню з метою ураження противником засобів протиповітряної оборони, про що свідчать численні випадки використання геолокаційних даних, що містяться в метаданих фотографій. Також важливо пам'ятати, що метадані не є просто "додатковою" інформацією. Вони можуть надати вкрай точне і детальне уявлення про наше життя, інтереси, звички і навіть місце проживання. Сюди входять всі фотографії, які ми публікуємо в соціальних мережах, всі аудіофайли, які ми прослуховуємо, і всі відео, які ми дивимося. Наприклад, наші журналісти вже ефективно використовують

метадані для викриття російських фейків. 19 лютого окупаційна влада на територіях ОРДЛО повідомила про замінування мосту біля міста Краснодон Луганської області. Залізничним мостом нібито мав пройти потяг із біженцями з окупованих територій, який і планували підірвати українські диверсанти. Проте журналісти українського видання Focus з'ясували, що відео, яке опублікували бойовики на доказ своєї версії, зняли три роки тому, чим вчергове довели відсутність будь-якої легітимності слів росіян [4].

Тому, розробка та вдосконалення засобів обробки метаданих для захищеної передачі мультимедійного трафіку стає **актуальною темою дослідження**. Вирішення цієї проблеми вимагає розробки нових методів та алгоритмів, які дозволять ефективно фільтрувати, шифрувати або приховувати компрометуючу інформацію, що міститься в метаданих мультимедійних файлів. Дослідження в цій області має великий потенціал для покращення захисту приватності та забезпечення безпеки при обміні мультимедійними даними.

Об'єкт дослідження - засоби обробки метаданих для захищеної передачі мультимедійного трафіку.

Предмет дослідження - конфіденційність особистих даних при передачі мультимедійного трафіку, що може бути порушена через наявність компрометуючої інформації в метаданих.

Методи дослідження:

- аналіз відкритих джерел;
- аналіз існуючих методів та технік обробки метаданих;
- аналіз механізмів захищеної передачі мультимедійного трафіку;
- розробка архітектури програмного модуля.

Практична цінність роботи полягає в реалізації програмного модуля для обробки та керуванням метаданих в мультимедійних файлах з можливістю відправки відредагованих мультимедійних файлів захищеним протоколом для забезпечення конфіденційності відправника.

РОЗДІЛ 1
МЕТАДАНІ В МУЛЬТИМЕДІЙНИХ ФАЙЛАХ, КЛАСИФІКАЦІЯ ТА СТАНДАРТИ

1.1. Визначення та класифікація метаданих

У сучасній інформаційній культурі широке використання цифрових фотографій повністю змінило спосіб створення, розповсюдження та збереження візуальних даних. Однак разом із зручністю та доступністю цифрових зображень виникає проблема захисту метаданих, пов’язаних із цими зображеннями.

Метадані — це «дані, які надають інформацію про інші дані», але не вміст даних, наприклад текст повідомлення чи саме зображення. Метадані збагачують дані інформацією, яка полегшує пошук, використання та керування ними.

Метадані фото – це інформація, корисна у звичному випадку, але небезпечна для тих, хто хоче забезпечити собі максимальну анонімність. Метадані є у кожної фотографії незалежно від того, з якого пристрою вона була зроблена. Файл зображення — будь то JPEG, TIFF, PSD, Raw чи кілька інших форматів — може містити ряд метаданих. Екосистема цифрових зображень залежить від метаданих, які складаються з даних, що містяться у файлах цифрових зображень, таких як марка та модель камери, дата й час збору та координати розташування.

Різні типи або класи текстової інформації про цифрові файли, які називаються метаданими, служать окремим цілям. Деякі класи метаданих можуть бути вбудованими у файли цифрових зображень. Деякі схеми або формати даних фактично ідентифікують свої елементи цими класами, хоча це може бути неочевидним. Кожен із наступних трьох «класів» метаданих стає частиною файлу зображення, якщо його вбудовано у JPEG, TIFF, PSD, Raw або кілька інших популярних форматів. Їх також можна зберігати у побічних файлах (Побічні файли, також відомі як файли друзів або підключені файли, — це комп’ютерні файли, які зберігають дані (часто метадані), які не підтримуються форматом вихідного файлу.

Для кожного вихідного файлу може існувати один або кілька додаткових файлів. Також можуть існувати «бази метаданих», де одна база даних містить метадані для кількох вихідних файлів) [5].

Технічні метадані

Більшість сучасних пристроїв захоплення зображень генерують інформацію про себе та зображення, які вони записують, наприклад, що зберігається в Exif. Ці дані описують технічні характеристики зображення, такі як його розмір, колірний профіль, швидкість ISO та інші налаштування камери. Деякі професійні камери можна налаштувати для додавання детальної інформації про право власності та описової інформації в поле примітки або коментаря, що зберігається в контейнері Exif [6].

Описові метадані

Фотограф або менеджер колекції зображень може вводити та вбудовувати різну інформацію про вміст зображення. Це може включати підписи, заголовки, ключові слова, місце зйомки тощо. Ці поля метаданих були включені в оригінальну схему IPTC-IIM. Їх було розширено в схемах метаданих IPTC Core і IPTC Extension.

Адміністративні метадані

Файли зображень також можуть включати умови ліцензування або використання прав, конкретні обмеження на використання зображення, випуски моделей, інформацію про походження, таку як особу творця, і контактну інформацію власника прав або ліцензіара. Ці типи метаданих були комплексно розглянуті та стандартизовані в системі PLUS. Схеми IPTC Core і IPTC Extension також розширюють типи та кількість такої інформації, яку можна зберігати в метаданих.

1.2. Стандартів та форматів метаданих у мультимедійних файлах

JPEG, TIFF, PSD, Raw і кілька інших форматів файлів можуть містити будь-який або всі стандартні типи метаданих

IPTC-IIM

Цю схему, яку часто називають «застарілою» IPTC, було розроблено на початку 1990-х років, головним чином для допомоги новинним організаціям у створенні підписів до ранніх цифрових зображень [7]. Його головна перевага полягає в тому, що більшість програм для редагування та керування зображеннями можуть читати та записувати його широко сумісні поля. Схема IPTC-IIM поклала початок і залишається широко використовуваним і визнаним форматом.

Незважаючи на те, що оригінальна схема Міжнародної ради з питань преси та телекомунікацій зараз вважається застарілим форматом, вона широко розпізнається програмними продуктами, які мають доступ до метаданих, багато з яких не можуть читати чи записувати нещодавно визначену схему IPTC Core/XMP.

Базуючись на форматі для текстових файлів, які описують супровідні медіа, IPTC Модель обміну інформацією(Information Interchange Model) (або скорочено IIM), запущена в 1991 році, забезпечуючи новий спосіб обробки «цифрових ресурсів» з метаданими та вмістом, що зберігаються в двійковій структурованій структурі. У середині 1990-х Adobe додала можливість вставляти описові метадані у файли цифрових зображень TIFF і JPEG, що породило заголовки IPTC [8].

Представлений IIM був розроблений для опису всіх типів медіа (статті, зображення тощо). Деякі з перших програм цифрової обробки зображень (наприклад, Photoshop) визнали це особливо корисним набором значень, і спільнота фотографів вподобала деякі з них. Після випуску IPTC Core у 2005 році це вже не був єдиний спосіб вбудовування метаданих фотографій. Багато новіших програм використовували метод XMP для зберігання метаданих, але зберігали зворотну сумісність із стандартом IIM шляхом синхронізації даних і запису збережених значень (для будь-яких спільних полів) у контейнери IIM і XMP.

Багато сторонніх розробників створили програми, які читають і записують у заголовки IPTC. Але деякі вважають метадані «чорною магією», оскільки місце зберігання та структура змінюються залежно від формату файлу зображення. Час від часу з’являються збої в тому, як певні діакритичні символи – такі як наголоси, тильди, тощо (áçèîñõü) – перекладаються під час обміну файлами між операційними

системами. Ця схема також має певні обмеження щодо кількості символів, які може містити кожне поле.

IPTC Core & Extension

Ця новіша схема базується на спадщині IIM, додаючи більше типів описової та адміністративної інформації, включаючи нові поля для задоволення потреб спільнот стокової фотографії та культурної спадщини, упакованих у більш надійний формат даних, "XMP" [9].

Основна схема IPTC додає панелі та поля в більш гнучкий, надійний формат даних.

У 2005 році Міжнародна рада з питань преси та телекомунікацій випустила оновлений стандарт використання даних IPTC у схемі Adobe XMP, який отримав назву «IPTC Core». Це дає змогу включати дані IPTC у більш широкий спектр форматів зображень, таких як JPEG, TIFF, JPEG2000, тощо.

Оскільки XMP підтримує текст Unicode, він може представляти нелатинські алфавіти (наприклад, кирилицю, японську, китайську). Це також вирішує проблему спотворення діакритичних символів під час переміщення зображень між операційними системами Macintosh і Windows. На відміну від застарілого формату, IPTC Core не має певних обмежень на символи для кожного поля, за винятком підтримки зворотної сумісності з оригінальною схемою IPTC.

PLUS

Універсальна система ліцензування зображень для ідентифікації та визначення ліцензій на використання зображень описує схему та інструменти для створення рядка символів, які можуть ідентифікувати власника авторського права, користувача, обсяг і умови використання ліцензованого зображення. PLUS надає універсальний стандарт метаданих для опису ліцензування та прав, наданих для фотографій [10].

Міжнародна некомерційна коаліція PLUS розробляє, затверджує та підтримує набір стандартів для мови та форматів ліцензування. Вона виступає як головна асоціація, що представляє видавців, дизайнерів, рекламні агенції, фотографів, ілюстраторів, розповсюджувачів зображень, представників художників, музеїв, бібліотек та органів стандартизації, такі як UPDIG, IPTC, IDEAlliance та інші.

Веб-сайт PLUS пропонує безкоштовні інструменти для вбудовування та читання ліцензій PLUS за допомогою формату метаданих XMP. Рядок ліцензії може міститися в метаданих IPTC або XMP із можливістю прямого вбудовування у файл зображення. Можливість легко ідентифікувати власника прав на зображення та зв'язатися з ним стане життєво важливою, коли запропоноване законодавство про «твори-сироти» або закони про реформу авторського права стануть законом. Вбудовування повних і точних метаданих прав за допомогою метаданих IPTC, IPTC Core та/або PLUS допомагає захистити зображення від цих та інших неліцензійних видів використання.

XMP

Це новий формат даних, який використовується IPTC Core and Extension для зберігання та доступу до метаданих зображень. Це дає змогу зберігати метадані у файлі зображення або в супровідному файлі додаткової інформації, а також дозволяє створювати власні поля метаданих [11]. XMP пропонує надійний, гнучкий, міжплатформний метод для зберігання метаданих зображень.

Платформа Extensible Metadata Platform або XMP — це певний тип розширюваної мови розмітки, який використовується для зберігання метаданих у цифрових фотографіях. Adobe представила формат у 2001 році, коли випустила Photoshop 7. У 2004 році Adobe, IPTC і IDEAlliance разом представили основну схему IPTC для XMP. Він переносить значення метаданих із заголовків IPTC у більш сучасний і гнучкий формат XMP. Унікальною перевагою XMP є те, що він дозволяє створювати власні панелі метаданих. Вони не лише зберігають додаткові форми даних, але й організовують їх інакше, ніж у Photoshop за замовчуванням «Інформація про файл». Ці панелі на основі XMP можна встановити у Photoshop і вони дозволяють будь-кому вставляти спеціальні метадані у файли зображень.

Проте, хоча ви можете додавати спеціальну інформацію таким чином, лише Adobe Photoshop і Bridge, а також кілька інших баз даних зображень можуть імпортувати або переглядати ці метадані. А спеціальні панелі потребують додаткового налаштування, перш ніж інші зможуть їх використовувати.

Exif

Ці метадані, які часто створюються камерами та іншими пристроями зйомки, включають технічну інформацію про зображення та метод його зйомки, наприклад налаштування експозиції, час зйомки, інформацію про місцезнаходження GPS і модель камери. Змінний формат файлу зображень зберігає технічні метадані про захоплення, характеристики зображення тощо.

Цифрові фотоапарати вбудовують технічні метадані, які називаються даними Exif, у файли зображень (переважно формати JPEG і TIFF), які вони створюють. Основною особливістю Exif є його здатність записувати інформацію про камеру у файл зображення в момент зйомки. Деякі загальні поля даних включають марку та модель камери, її серійний номер, дату та час зйомки зображення, витримку, діафрагму, використовуваний об’єктив і налаштування швидкості ISO. Метадані Exif часто включають інші технічні деталі, такі як баланс білого та відстань до об’єкта [12].

Dublin Core

Багато бібліотек зображень і різноманітні галузі зберігають інформацію з файлами зображень за допомогою цієї схеми. Кілька його полів сумісні з форматами IPTC. Dublin Core — це проста, загальна, широко адаптована схема метаданих.

Названа на честь Дубліна, штат Огайо, де бібліотекарі вперше обговорили її потребу, схема зараз підтримується Dublin Core Metadata Initiative (DCMI). DC складається з 15 основних елементів. NISO (Національна організація стандартів інформації) та ISO (Міжнародна організація стандартів) прийняли його як стандарт. Поточна схема IPTC Core містить п’ять полів, сумісних із Dublin Core (Назва, Тема/Ключові слова, Творець, Повідомлення про права/авторські права, Опис) [13].

1.3. Законодавча база з питань конфіденційності інформації, яка зберігається у метаданих мультимедійних файлів

Розглядаючи питання конфіденційності інформації яка зберігається у метаданих мультимедійних файлів можна згадати декілька нормативно-правових документів. Одним із них є Закон України "Про захист персональних даних" [14] який

встановлює правові норми, які регулюють збирання, обробку, зберігання та передачу персональних даних, включаючи ті, які можуть міститися у метаданих мультимедійних файлів. Основними положеннями закону, що стосуються метаданих є визначення персональних даних.

Закон визначає широке поняття персональних даних, яке охоплює будь-яку інформацію, що стосується ідентифікованої або можливо ідентифікувати особи. Закон також визначає конфіденційну інформацію, включаючи дані про фізичну особу, та встановлює, що така інформація може бути передана лише за згодою або за бажанням відповідної особи. Це безпосередньо впливає на обробку метаданих в інформаційній системі, оскільки вони можуть містити персональні дані користувачів, географічні дані, дати, час, моделі пристроїв, авторство та іншу інформацію. Зокрема, якщо метадані містять таку інформацію, їх можна вважати конфіденційною інформацією, яку необхідно належним чином захищати.

Закон встановлює принципи та обмеження, які необхідно дотримувати при обробці персональних даних, включаючи принципи легальності, обмеження цілей, точності, обмеження зберігання та інші. У контексті метаданих, обробка повинна відбуватися з дотриманням цих принципів. Закон також накладає обмеження на передачу персональних даних третім особам без згоди суб'єкта даних. Ці обмеження також поширюються на передачу метаданих мультимедійних файлів, якщо такі дані вважаються персональними. Цим законом визначено, що власник даних має право на доступ до власних даних, право на виправлення, видалення даних, право відмовитися від обробки даних та інше. В контексті метаданих мультимедійних файлів суб’єкт має повне право на видалення всіх можливих метаданих – місце розташування, де було здійснено фотографія або відео, ім’я автора файлу, назву пристрою за допомогою якого було зроблено файл та інше, їх зміну для збереження конфіденційності.

Додатково, існує ще один закон, до якого можна звернутися. Особливо варто відзначити Закон України "Про інформацію" [15], який встановлює правила доступу, розповсюдження та захисту інформації. Обробка метаданих тісно пов'язана з використанням інформації.

Цей закон також застосовується до метаданих, які є структурованою інформацією про дані, які містять різні типи інформації. Визначення суб'єктів та об'єктів інформаційних відносин є важливим елементом цього закону. У контексті метаданих, суб'єктами можуть бути користувачі, які створюють або використовують дані, організації, які збирають, зберігають або аналізують метадані, а також провайдери послуг, які надають платформи або інструменти для обробки метаданих. Об'єктом цих відносин є самі метадані. Відповідно до Закону України "Про інформацію", метадані мультимедійних файлів, що знаходяться у володінні державних органів, органів місцевого самоврядування, підприємств, установ та організацій, можуть бути доступними громадянам. Однак розповсюдження таких даних може бути обмеженим, особливо якщо метадані містять конфіденційну інформацію. Закон також вимагає захисту цих даних від несанкціонованого доступу, розкриття та використання.

Одним з основних законів Європейського Союзу, що регулює метадані мультимедійних файлів у контексті приватності, є Загальний регламент з охорони даних (General Data Protection Regulation, GDPR) [16].

GDPR є нормативно-правовим актом, що набрав чинності 25 травня 2018 року, і він має за мету захищати приватні дані громадян Європейського Союзу. Цей регламент створений для забезпечення контролю над збиранням, обробкою та передачею особистих даних, включаючи метадані, з метою забезпечення конфіденційності та приватності користувачів.

Згідно з GDPR, метадані, які можуть бути пов'язані з ідентифікованими або ідентифікованими фізичними особами, вважаються особистими даними і підпадають під регулювання цього законодавства.

1.4 Обробка метаданих застосунками для передачі мультимедіа трафіку

У статті [17] де я виступаю одним із авторів я та мої колеги провели дослідження, де визначили як застосунки для передачі мультимедіа обробляють

метадані при передачі їх по відкритих каналах зв’язку, та визначили загрози аспектів захисту інформації які можуть виникнути. Визначилось що численні популярні програми обміну повідомленнями автоматично стискають і витягують метадані із зображень, щоб зменшити розмір файлу та збільшити швидкість передачі, що може призвести до втрати важливої інформації, вбудованої в зображення.

Потенційна втрата автентичності та цілісності є одним із ризиків передавання зображень через програми обміну онлайн-повідомленнями. Без метаданих важко перевірити автентичність зображення, включаючи інформацію про камеру, налаштування та час зйомки. Це, може бути особливо проблематичним у юридичних або криміналістичних контекстах, де автентичність зображення має вирішальне значення, наприклад, у накопиченні доказів або суперечках щодо інтелектуальної власності. Крім того, відсутність метаданих полегшує зловмисникам маніпуляції або зміну зображень, що може призвести до поширення дезінформації або неправдивих зображень.

Ще одна небезпека – втрата особистої приватності. Як згадувалося раніше, метадані можуть містити конфіденційну інформацію, таку як дані про геолокацію або ідентифікаційну інформацію, яка може бути ненавмисно розкрита, коли зображення поширюються через програми обміну повідомленнями. Це може оприлюднити місцезнаходження, діяльність або особу користувачів без їхньої згоди, створюючи значний ризик конфіденційності. Це особливо непокоїть у випадках, коли люди обґрунтовано сподіваються на конфіденційність, наприклад, з особистими або конфіденційними зображеннями.

Крім того, безпека метаданих цифрових зображень може бути порушена під час передачі через програми обміну повідомленнями. Вміст повідомлень може бути зашифрований програмами обміну повідомленнями, але не метадані, залишаючи їх чутливими до перехоплення, несанкціонованого доступу або зміни. Це може призвести до порушення конфіденційності, витоку даних та інших інцидентів безпеки, які можуть завдати шкоди репутації або юридичної відповідальності для користувачів і організацій. Щоб зменшити ризики передачі зображень через програми обміну повідомленнями, користувачі повинні знати про обмеження передачі

зображень через програми обміну повідомленнями та вжити необхідних заходів для захисту метаданих. Це може включати використання альтернативних методів, наприклад електронної пошти або хмарних служб зберігання, які зберігають метадані, шифрування зображень перед їх передачею або використання програм, які надають пріоритет збереженню та безпеці метаданих. При цьому сервіси електронної пошти або хмарних служб зберігання повинні чітко інформувати користувачів відносно того, що окрім змісту зображення вони передають усі доступні метадані які є у зображення, а також надати механізм їх зручного видалення, для забезпечення конфіденційності користувача.

1.5 Типові загрози конфіденційності, пов'язані з метаданими

За розвитку інформаційних технологій, коли мультимедійні файли стають основними засобами передачі інформації, проблеми забезпечення конфіденційності стають все більш актуальними. Однією з основних причин цих проблем є метадані - важлива складова частина мультимедійних файлів. Втрата контролю над метаданими може призвести до розкриття конфіденційної інформації і, в результаті, підірвати особисту безпеку та конфіденційність. Зловмисники можуть скористатися цим для отримання особистої інформації про людину, яка навіть не усвідомлює, що разом із своїми фото або відео передає дані про місцезнаходження під час зйомки або модель пристрою, на якому зроблено фото або відео. Це може надати зловмисникам можливість розробити шкідливе програмне забезпечення для подальшого стеження або вимагання грошей.

Проблема з метаданими полягає в тому, що вони часто створюються і зберігаються без належного відома користувача. Це призводить до потенційних загроз конфіденційності, таких як:

- Розкриття геолокації та особистої інформації
- Виток технічних характеристик

1.5.1 Розкриття геолокації та особистої інформації

Метадані, які містяться в мультимедійних файлах, можуть становити значну загрозу конфіденційності користувачів. Вони містять різноманітну інформацію, яка може бути використана для небажаного доступу до особистої інформації користувача або її зловмисного використання.

Наприклад, обробка фотографій або інших мультимедійних файлів, таких як відео або аудіо, може викрити метадані як джерело витоку інформації. Багато сучасних камер, включаючи ті, що вбудовані в смартфони, автоматично додають до фотографій EXIF-дані. Ці дані містять інформацію про умови зйомки, такі як час зйомки і параметри експозиції, а також можуть містити геотеги - інформацію про місцезнаходження зйомки. Якщо такі фотографії опубліковані в Інтернеті без відповідної обробки, ці дані стають доступними для всіх, хто має доступ до фотографії.

Таким чином, розкриття даних про місцезнаходження в метаданих мультимедійних файлів є однією з найбільш критичних загроз конфіденційності. Геолокаційні метадані містять інформацію про фізичне місце створення або модифікації файлу, включаючи географічні координати, висоту, швидкість, напрямок та інші геопросторові дані [18]. Ці дані можуть дати зловмисникам достатньо інформації для виявлення шаблонів поведінки користувача, таких як часті місця перебування. Знання цих шаблонів поведінки може бути використане зловмисниками для шахрайства, крадіжок або навіть фізичних загроз, особливо для осіб з публічним статусом.

Крім того, отримання інформації про місцезнаходження під час фото або відеозйомки може бути використане ворогом у воєнних конфліктах. Наприклад, під час вторгнення Росії до України у 2022 році, російські силовики використовували фото та відеоматеріали з соціальних мереж для корегування своїх цілей. Багато людей не усвідомлюють, що викладаючи мультимедійні файли в Інтернет або соціальні мережі, вони також розкривають інформацію про місцезнаходження техніки або важливих військових об'єктів [19].

Метадані місцезнаходження можуть розкривати конкретні події, в яких брав участь користувач. Ця інформація може використовуватися зловмисниками для встановлення зв'язків між людьми, що може призвести до витоку особистої інформації про взаємини або професійні стосунки. Наприклад, якщо на певній події було багато людей, і один користувач опублікував фотографію чи відео з цього заходу, зловмисник може використовувати метадані місцезнаходження, щоб знайти інші фото та відео з цієї події. Потім, за допомогою спеціального програмного забезпечення, він може знайти імена людей, що були присутні на цій події, і використовувати цю інформацію для отримання багатьох інших особистих даних. Ця інформація може бути використана для шантажу або навіть фізичного насилля.

- В цілому, розкриття місцезнаходження метаданими призводить до таких загроз конфіденційності користувачів:

- Встановлення особи: Ненавмисне розкриття геолокаційних метаданих може призвести до виявлення особистості користувача, особливо якщо користувач прагне залишатися анонімним.

- Розкриття місця проживання: Постійне публікування контенту з конкретної локації може непрямо вказувати на адресу проживання користувача, що може бути проблемою, якщо користувач не хоче розголошувати цю інформацію.

- Визначення життєвих звичок: Регулярне публікування контенту з певних місць може допомогти іншим зрозуміти повторювані моделі поведінки користувача. Наприклад, якщо користувач часто ділиться контентом з певної кав'ярні, це може свідчити про його звичку відвідувати це місце.

- Використання третіми сторонами: Геолокаційні дані можуть бути використані третіми сторонами для різних цілей, включаючи маркетинг і стеження, що може порушувати приватність користувача.

Ім'я користувача, яке приховане в метаданих, може стати критичним вихідним пунктом для проведення цілеспрямованих атак. Термін "цілеспрямована атака" відноситься до специфічної взаємодії зловмисника з жертвою, при якій атакується конкретна особа або організація. В рамках соціальної інженерії, зловмисники можуть використовувати знайдене в метаданих ім'я як початкову точку для збору додаткової

інформації. Це може включати пошук на соціальних мережах, де багато людей публікують свої особисті дані, що допомагає знайти більше особистої інформації, такої як дата народження, адреса, робота, хобі тощо. Також можливий пошук даних про особу на форумах або інших платформах, де вона може брати участь.

Отримана інформація може використовуватися для підготовки фішингових атак, які виглядають більш достовірно, оскільки містять конкретні деталі, відомі жертві. Якщо метадані містять контактну інформацію, таку як електронна адреса або номер телефону, зловмисники можуть безпосередньо зв'язатися з жертвою. Це може призвести до різних форм шахрайства, наприклад, підроблених повідомлень від "банку" або "постачальника послуг", які вимагають від жертви надати додаткові дані або ввести свої особисті дані на підробленому веб-сайті. Крім того, зловмисники можуть спробувати маніпулювати жертвою, створюючи враження термінової потреби або критичної ситуації, в якій жертва повинна швидко відгукнутися, знижуючи рівень пильності. У крайньому випадку зловмисники можуть використовувати особисту інформацію для цілком недемократичних форм впливу, таких як погрози або шантаж. Наприклад, зловмисник може твердити, що володіє компрометуючою інформацією про жертву або що може завдати шкоди жертві або її близьким, якщо жертва не виконає певні вимоги.

1.5.2 Викриття технічних характеристик пристроїв

Розкриття технічних характеристик пристроїв, з яких створюються мультимедійні файли через несанкціонований доступ до метаданих може створювати серйозні проблеми з конфіденційністю. Ця проблема полягає в тому, що можуть бути виявлені технічні деталі пристрою, на якому було створено або редаговано цей файл. Такі дані можуть включати модель камери, версію програмного забезпечення, налаштування системи та унікальні ідентифікатори обладнання, а також інші технічні параметри. Зловмисники можуть використовувати цю інформацію для проведення цілеспрямованих атак на конкретне програмне забезпечення або обладнання.

Розкриття технічних характеристик може збільшити ризики для людини на декількох рівнях.

По-перше, це надає зловмисникам допомогу у виявленні потенційних вразливостей, які відомі для певних моделей обладнання або версій програмного забезпечення, та їх подальшого використання у системі жертви. Знання про це дозволяє зловмисникам відносно легко реалізувати атаки з метою досягнення бажаного результату.

По друге, зловмисники можуть використовувати методи OSINT для знаходження компрометуючої інформації, включаючи мультимедійні файли, які належать жертві. Після знаходження цих файлів вони аналізують метадані, що зберігаються в них, щоб виявити операційну систему, якою користується жертва, моделі пристроїв, які вона використовує, та інші технічні дані. Зловмисники можуть використовувати цю інформацію для розробки шкідливого програмного забезпечення для стеження за жертвою і отримання більш детальної інформації або для створення програм вимагачів з метою вимагання грошових винагород. Після цього зловмисник може використати знайдену інформацію про технічні характеристики для створення більш переконливих шахрайських схем або соціальної інженерії. Наприклад, зловмисник може створити електронний лист, який видаватиметься за офіційне повідомлення від виробника камери, яку використовує користувач, і в цьому листі запропонувати користувачеві надати особисту інформацію або завантажити "оновлення" програмного забезпечення, яке фактично є шкідливим ПЗ, створеним зловмисником.

1.6 Сучасні методів та техніки обробки метаданих для забезпечення конфіденційності

У контексті сучасних технологій інформаційної безпеки, обробка метаданих є надзвичайно важливим елементом для забезпечення конфіденційності, особливо в випадку мультимедійних файлів. Одним із ключових напрямків у роботі з метаданими є їх анонімізація та псевдонімізація [20, 21].

Анонімізація включає процес видалення або заміни ідентифікуючої інформації в метаданих, тоді як псевдонімізація передбачає заміну ідентифікуючих даних на синтетичні значення, які неможливо прямо пов'язати з реальною особою. Анонімізація метаданих часто використовується для зменшення ризику витоку конфіденційної інформації. Це може означати видалення таких даних, як ім'я користувача, контактна інформація та інші елементи, що можуть розкривати особистість. Однак анонімізація не є повністю непроникною. У певних обставинах зловмисники можуть спробувати відновити анонімізовану інформацію шляхом використання методів де-анонімізації.

Псевдонімізація метаданих є ще одним методом, часто використовуваним для забезпечення конфіденційності. Цей підхід передбачає заміну особистої інформації на псевдоніми, які не прямо вказують на реальну особу. Застосування цього методу дозволяє захистити особисті дані.

Крім анонімізації та псевдонімізації, існують інші технології, які можуть бути застосовані для захисту метаданих, включаючи шифрування та управління доступом. Шифрування допомагає забезпечити захист метаданих від несанкціонованого доступу та витоку. Управління доступом дозволяє контролювати, хто має право переглядати та змінювати метадані.

Машинне навчання та штучний інтелект також можуть бути ефективними інструментами для автоматичної обробки метаданих в мультимедійних файлах. Зокрема, ці технології можуть використовуватися для автоматичного виявлення та видалення чутливої інформації, що міститься в метаданих.

У випадку мультимедійних файлів, захист метаданих потребує особливого підходу. Оскільки метадані можуть містити значну кількість інформації, яка не повинна бути доступною зловмиснику, їх захист від витоку та недоброзичливого використання є важливим завданням. Для ефективного захисту метаданих в мультимедійних файлах можна використовувати комбінацію різних методів та технологій. Це може включати використання різних методів анонімізації та псевдонімізації, спільно зі штучним інтелектом, шифруванням метаданих або встановленням жорсткого контролю доступу до метаданих.

Найкращим способом зрозуміти, як саме працюють ці методи обробки метаданих для забезпечення конфіденційності, є розгляд прикладів. Наприклад, для пояснення методів анонімізації та псевдонімізації можна взяти фотографію, зроблену на смартфоні, яка містить метадані EXIF з точною геолокацією знімку. Процес анонімізації в цьому контексті включатиме видалення даних геолокації з метаданих за допомогою спеціального програмного забезпечення або веб-ресурсів, які можуть виконати цю операцію. Щодо псевдонімізації, можна замінити точні координати загальною локацією, наприклад, містом або країною. Варто враховувати, щоб уникати надання схожих на правдиві дані, тому у разі зміни інформації про місцезнаходження під час зйомки, кращим рішенням буде не вказувати сусідні вулиці або райони.

Шифрування є ефективним способом захисту будь-якої інформації, включаючи метадані. Наприклад, якщо ми візьмемо відеофайл з метаданими, які містять дані про автора, використане програмне забезпечення та дату його створення, шифрування цих метаданих забезпечить доступ до них тільки особам з відповідним ключем дешифрування. Таким чином, шифрування можна комбінувати з методами псевдонімізації, щоб максимально забезпечити безпеку автора відео, яке він завантажив в Інтернет.

Управління доступом також може бути використано для контролю доступу до метаданих. Наприклад, в разі pdf- або doc-файлу, що містить автоматично згенеровані метадані про автора, дату створення та внесені зміни, використання методів управління доступом може обмежувати, хто може переглядати або змінювати ці метадані. Ви можете налаштувати файл таким чином, що тільки автор документа матиме доступ до метаданих або може навіть заборонити перевірку вмісту метаданих.

Машинне навчання та штучний інтелект є популярними напрямками, які швидко розвиваються у сучасному світі. Вони можуть бути використані для обробки метаданих з метою забезпечення конфіденційності. Застосування цих технологій дозволяє виявляти та видаляти чутливу інформацію, що міститься в метаданих. Наприклад, штучний інтелект, навчений за допомогою машинного навчання, може

аналізувати метадані зображень, відео та інших мультимедійних файлів, щоб виявляти та видаляти інформацію про особу, яка створила або редагувала ці файли.

Висновок до 1 розділу

У цьому розділі була розглянута законодавча база України та інших країн світу, яка регулює розповсюдження, використання, зберігання та збирання метаданих. Серед законів, що регулюють ці питання, варто відзначити Закон України "Про інформацію", Закон України "Про захист персональних даних" і регламент Європейського Союзу щодо захисту персональних даних (GDPR).

Була розглянута класифікація метаданих, яка включає такі типи:

- Описові метадані, що містять інформацію, наприклад, контактні дані, назву або автора публікації, резюме роботи, ключові слова, географічне положення або пояснення методології.
- Технічні метадані, що описують, описують технічні характеристики зображення, такі як його розмір, колірний профіль, швидкість ISO та інші налаштування камери.
- Адміністративні метадані, які можуть включати дати створення або надходження, права доступу, походження або правила використання, такі як зберігання або видалення.

Основними стандартами для мультимедійних файлів є:

- IPTC-IIM
- PLUS
- XMP
- IPTC Core & Extension
- EXIF
- Dublin Core

Основними проблемами безпеки конфіденційності метаданих є:

- Розкриття геолокації та особистої інформації, що включає розкриття інформації про місцезнаходження під час зйомки фотографій або відеозаписів, а також викриття імені автора файлу.
- Розкриття технічних характеристик, таких як версія програмного забезпечення, тип пристрою тощо, що дозволяє зловмиснику легше визначитися з типом атаки, такої як соціальна інженерія або поширення шкідливого програмного забезпечення, спрямованого на жертву.

Серед сучасних методів та технік обробки метаданих можна виділити:

- Анонімізацію та псевдонімізацію, які використовуються для видалення або модифікації чутливої інформації.
- Шифрування інформації, що міститься в метаданих.
- Управління доступом, яке встановлює обмеження щодо того, хто і що може отримати з метаданих.
- Використання машинного навчання та штучного інтелекту для швидкого аналізу, виявлення та видалення чутливої інформації, що міститься в метаданих мультимедійних файлів.

РОЗДІЛ 2
ПРОТОКОЛИ ПЕРЕДАЧІ МУЛЬТИМЕДІЙНИХ ФАЙЛІВ У МЕРЕЖІ ІНТЕРНЕТ

2.1 Визначення прицнипів передачі мультимедійного трафіку на прикладі протоколу FTP

Захист конфіденційності мультимедійних файлів має на увазі, що мультимедійні файли має отримати стороння особа. У сучасному світі обмін інформацією здійснюється в мережі Інтернет, нерідко по відкритим каналам зв’язку. З метою забезпечення конфіденційності метаданих необхідно розглянути протоколи та стандарти які дозволять безпечно обмінюватися інформацією, зі збереженням оригінальних метаданих які надсилаються під час інформаційного обміну. Основні принципи передачі файлів в мережі Інтернет включають в себе використання різних протоколів та форматів передачі. Для обміну мультимедійним трафіком зазвичай використовується проста схема передачі – файл спочатку кодується у новий формат після чого передається одним із протоколів обміну інформації. Розглянемо один з найпоширеніших способів передачі файлів - це використання протоколу передачі файлів (FTP).

FTP (File Transfer Protocol) - це протокол на прикладному рівні мережі OSI, який використовується для передачі файлів по мережі. Він з'явився ще в 1971 році, задовго до HTTP і навіть TCP/IP, і є одним з найстаріших прикладних протоколів. Протокол FTP описаний в RFC 959 [22]. Під час передачі, сервер відправляє потік управління у вигляді тризначних ASCII-кодів стану, додатково супроводжуючи їх необов'язковим текстовим повідомленням. Наприклад, "200" (або "200 OK") означає успішне виконання останньої команди. Цифри виступають як код відповіді, а текст надає роз'яснення або запит. Поточна передача даних може бути припинена за допомогою повідомлення про розрив з'єднання, яке надсилається по потоці управління. Однією з особливостей протоколу FTP є використання багатьох (принаймні, двох) з'єднань. За замовчуванням, FTP використовує два порти - 20 та 21, для передачі даних і команд

відповідно (див. рис. 1). Порт 20 є портом для передачі даних між клієнтом та потоком даних сервера. Порт 21 - це командний порт, який використовується для передачі потоку управління та обробки FTP-команд та параметрів, введених клієнтом. У межах однієї FTP-сесії можна одночасно передавати кілька файлів в обох напрямках. Для кожного каналу даних відкривається окремий порт TCP, номер якого вибирається або сервером, або клієнтом, залежно від режиму передачі

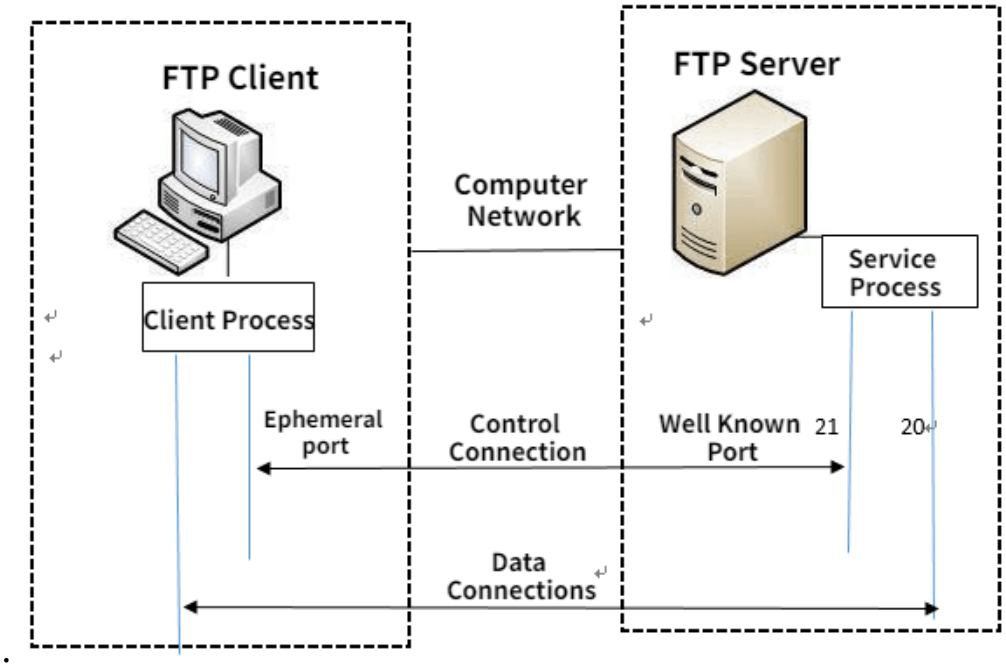

Рисунок 2.1 – Принцип роботи FTP протоколу.

FTP може працювати в двох режимах: активному і пасивному, і вибір режиму визначає спосіб встановлення з'єднання. У активному режимі клієнт створює керуюче TCP-з'єднання з сервером і надсилає свою IP-адресу та довільний номер порту клієнта серверу. Потім клієнт очікує, поки сервер запустить TCP-з'єднання з вказаною IP-адресою та номером порту. Однак, у випадку, якщо клієнт використовує брандмауер і не може приймати вхідні TCP-з'єднання, застосовується пасивний режим. В пасивному режимі клієнт використовує потік управління, щоб надіслати серверу команду PASV, після чого від сервера отримує IP-адресу і номер порту, які клієнт використовує для встановлення з'єднання для потоку даних з будь-якого доступного клієнтського порту до отриманої адреси та порту. Обидва режими були оновлені у вересні 1998 року для підтримки IPv6. На той час також були внесені подальші зміни в пасивний режим, що призвели до розширеного пасивного режиму. При передачі

даних через мережу при використанні FTP протоколу можуть бути використані чотири типи представлення даних:

- ASCII – використовується для тексту. Дані, якщо необхідно, до передачі конвертуються з символьного представлення на пристрої відправнику в «восьмибітний ASCII», і за необхідності, у символьне представлення приймаючого хоста. Зокрема, змінюються символи перекладу рядка. (CR /chr(13)/, LF /chr(10)/ в Windows на LF /chr(10)/ в Unix/Linux). Як наслідок, цей режим не підходить для файлів, що містять не лише звичайний текст.
- Бінарний режим(або режим зображення) - пристрій-відправник посилає кожен файл байт за байтом, а одержувач зберігає потік байтів під час отримання. Підтримка цього режиму була рекомендована у всіх реалізаціях FTP.
- EBCDIC — використовується передачі звичайного тексту між хостами в кодуванні EBCDIC. В іншому цей режим аналогічний до ASCII-режиму.
- Локальний режим — дозволяє двом комп'ютерам з ідентичними налаштуваннями надсилати дані у власному форматі без конвертації в ASCII.

Окремо для текстових файлів надано різні формати керування та налаштування структури запису. Ці особливості були розроблені для роботи з файлами, які містять Telnet або форматування ASA. Включає в себе

- Поточний режим
- Блоковий режим
- Режим стиснення.

2.2 Огляд можливостей передачі мультмедійного трафіку з використанням протоколу HTTP

Протокол HTTP (Hypertext Transfer Protocol), початково описаний в RFC 2068 [23], є більш новим і широко використовуваним протоколом для передачі гіпертекстових документів в Інтернеті. HTTP є протоколом прикладного рівня, який розроблений для передачі даних у форматі гіпертекстових документів, основною формою яких є HTML. Проте, у сучасних часах його використовують для передачі

різноманітних типів даних. HTTP базується на моделі "клієнт-сервер", аналогічній до TCP. Це означає, що для встановлення такого з'єднання потрібні споживачі (клієнти), які ініціюють з'єднання та надсилають запити, а також постачальники (сервери), які очікують з'єднання, обробляють запити та повертають результати.

У HTTP основним об'єктом маніпуляції є ресурс, який ідентифікується URI (Uniform Resource Identifier) у запитах клієнта. Зазвичай ресурсами є файли, що зберігаються на сервері, але вони можуть також представляти логічні об'єкти або абстрактні сутності. Протокол HTTP має особливість, що він дозволяє вказувати спосіб представлення ресурсу за допомогою різних параметрів, таких як формат, кодування, мова тощо. Це досягається за допомогою HTTP-заголовків. Незважаючи на те, що сам протокол HTTP є текстовим, завдяки можливості вказівки способу кодування повідомлень, клієнт і сервер можуть обмінюватися двійковими даними.

HTTP є протоколом прикладного рівня, подібним до FTP і SMTP. В обміні повідомленнями використовується типова схема "запит-відповідь". Для ідентифікації ресурсів HTTP використовуються глобальні URI. У відміну від інших протоколів, HTTP не зберігає свого стану, що означає відсутність збереження проміжного стану між запитами та відповідями. Компоненти, що використовують HTTP, можуть самостійно зберігати інформацію про стан, пов'язану з попередніми запитами та відповідями (наприклад, "куки" на стороні клієнта, "сесії" на стороні сервера). Браузер, який надсилає запити, може відстежувати затримки відповідей, а сервер може зберігати IP-адреси та заголовки запитів останніх клієнтів. Проте сам протокол HTTP не має внутрішньої підтримки стану і не усвідомлює попередні запити та відповіді.

За допомогою HTTP можна завантажувати файли різних типів, включаючи текстові файли, зображення, відео та інші. При передачі файлу через HTTP, файл може бути вкладеним в запит або відповідь HTTP. У випадку доступу до даних через FTP або інші файлові протоколи, тип файлу (або тип даних, що містяться в ньому) визначається за розширенням імені файлу, що не завжди є зручним. Однак, перед передачею даних HTTP передає заголовок "Content-Type: тип/підтип", що дозволяє клієнту однозначно визначити, як обробляти отримані дані.

2.3 Аналіз можливостей передачі мультимедійного трафіку через електронну пошту

Також існує можливість передавати файли через електронну пошту в Інтернеті. Файли, що додаються до електронного листа, зазвичай кодуються і відправляються як частини повідомлення. Електронна пошта (або e-mail) - це спосіб передачі та отримання повідомлень за допомогою електронних пристроїв. Електронна пошта широко поширена і широко використовується як засіб комунікації; в багатьох сферах життя, таких як бізнес, торгівля, уряд, освіта, розваги та інші, адреса електронної пошти часто вважається основною і необхідною складовою.

Електронна пошта працює в комп'ютерних мережах, зокрема в Інтернеті, а також у локальних мережах. Сучасні системи електронної пошти базуються на моделі зберігання та пересилання повідомлень. Сервери електронної пошти приймають, пересилають, доставляють і зберігають повідомлення. Електронна пошта зазвичай працює за допомогою трьох стандартних протоколів: POP3, IMAP і SMTP.

2.3.1 Огляд поштового протоколу POP

POP (Протокол пошти) - це стандартний Інтернет-протокол прикладного рівня, який використовується клієнтами електронної пошти для отримання листів зі поштового сервера. Версія POP3 [24] є найпоширенішою та разом з IMAP є найбільш використовуваним протоколом для отримання електронної пошти.

Протокол поштового відділення надає клієнтським програмам користувача доступ через мережу Інтернет-протоколу (IP) до поштової скриньки (maildrop), яка зберігається на поштовому сервері. Цей протокол підтримує операції завантаження та видалення повідомлень. Клієнти POP3 підключаються, отримують всі повідомлення, зберігають їх на своєму комп'ютері та потім видаляють з сервера. Такий дизайн протоколу POP та його процедур був створений для задоволення потреб користувачів, які мають тимчасове підключення до Інтернету, наприклад, комутований доступ. Це дозволяло користувачам отримувати електронну пошту під

час підключення та потім переглядати та обробляти отримані повідомлення в автономному режимі.

Клієнти POP3 також можуть залишати листи на сервері після завантаження. Сервер POP3 прослуховує відомий порт номер 110 для обробки запитів. Зашифроване з'єднання для протоколу POP3 може бути запитане після ініціації протоколу за допомогою команди STLS, якщо підтримується, або за допомогою POP3S, який підключається до сервера через захищений транспортний рівень (TLS) або рівень захищених сокетів (SSL) на відомому TCP-порті номер 995. Протокол POP4 існує лише як неофіційна пропозиція, яка додає базове керування папками, підтримку багатокомпонентних повідомлень та керування прапорцями повідомлень для конкуренції з IMAP. Проте, його розвиток не продовжується з 2003 року.

2.3.2 Визначення особливостей протоколу отримання даних IMAP

Протокол IMAP (Internet Message Access Protocol) - це стандартний Інтернет-протокол, який використовується клієнтами електронної пошти для отримання повідомлень з поштового сервера через з'єднання TCP/IP. Докладний опис IMAP міститься в RFC 9051 [25]. IMAP розроблено з метою надання повного керування електронною скринькою з декількома клієнтами електронної пошти, тому зазвичай клієнти залишають повідомлення на сервері, поки користувач не видалить їх явно. Сервер IMAP зазвичай прослуховує порт номер 143. IMAP через SSL/TLS (IMAPS) використовує порт номер 993. Практично всі сучасні клієнти та сервери електронної пошти підтримують IMAP, який разом з протоколом POP3 (Post Office Protocol) є двома найпоширенішими стандартними протоколами для отримання електронної пошти. Багато постачальників веб-пошти, таких як Gmail і Outlook.com, також підтримують як IMAP, так і POP3.

На відміну від протоколу POP3, який зазвичай з'єднується з сервером електронної пошти на короткий час для завантаження нових повідомлень, клієнти, які використовують IMAP4, часто залишаються підключеними, поки активний інтерфейс користувача, і завантажують вміст повідомлень за потребою. Цей шаблон

використання IMAP4 дозволяє швидшу відповідь для користувачів, які мають багато або великі повідомлення. Після успішної автентифікації протокол POP забезпечує статичне уявлення про поточний стан поштової скриньки та не надає механізму для відображення зовнішніх змін стану під час сеансу. Натомість, протокол IMAP забезпечує динамічний перегляд зовнішніх змін в стані поштової скриньки, включаючи нещодавно надіслані повідомлення та зміни, внесені іншими одночасно підключеними клієнтами.

Зазвичай електронна пошта в Інтернеті передається у форматі MIME, що дозволяє повідомленням мати ієрархічну структуру, де листові вузли можуть бути різних типів вмісту, а нелистові вузли можуть бути складними типами з багатьма компонентами. Протокол IMAP4 дозволяє клієнтам отримувати окремі частини повідомлення MIME, а також отримувати блоки окремих частин повідомлення. Ці механізми дозволяють клієнтам отримувати текстову частину повідомлення без вкладених файлів або передавати вміст повідомлення поступово під час отримання.

За допомогою прапорців, визначених у протоколі IMAP4, клієнти можуть відстежувати стан повідомлення, такий як прочитано, відповідь на нього або видалення. Ці позначки зберігаються на сервері, тому різні клієнти, які отримують доступ до однієї поштової скриньки в різний час, можуть бачити зміни стану, зроблені іншими клієнтами. Протокол POP не має механізму для збереження такої інформації про стан на сервері, тому якщо користувач отримує доступ до поштової скриньки з двома різними POP-клієнтами (в різний час), інформація про стан, наприклад, прочитано або доступ до повідомлення, не може бути синхронізована між клієнтами.

2.3.3 Аналіз принципів роботи протоколу SMTP та його розширених версій

Протокол передачі електронної пошти Simple Mail Transfer Protocol (SMTP) є стандартним засобом комунікації в Інтернеті для обміну електронними повідомленнями. SMTP використовується поштовими серверами та іншими агентами передачі повідомлень для надсилання й отримання електронних листів. Клієнти електронної пошти на рівні користувача зазвичай використовують SMTP для

надсилання повідомлень на поштовий сервер з метою подальшої пересилки. Зазвичай вони використовують порти 587 або 465 відповідно до RFC 8314 [26] для надсилання вихідної пошти на поштовий сервер. За своєю функціональністю, протоколи IMAP та POP3 використовуються для отримання електронної пошти, тоді як SMTP використовується для відправки.

SMTP з'явився в 1980 році, базуючись на раніше реалізованих концепціях в мережі ARPANET, яка була запущена в 1971 році. Протокол був оновлюваний, змінювався та розширювався впродовж часу. Сучасна версія протоколу має гнучку структуру з різними розширеннями, що дозволяють автентифікацію, шифрування, передачу бінарних даних та використання інтернаціоналізованих адрес електронної пошти.

Оригінальний SMTP

У 1980 році Джон Постел і Сюзанна Слюйзер запропонували використання протоколу передачі пошти як альтернативи FTP для обміну листами. Пізніше, у листопаді 1981 року, Постел опублікував RFC 788 [27] з назвою "Простий протокол пересилання пошти". Протокол SMTP був розроблений приблизно в той же час, коли і мережа Usenet, яка була заснована на моделі "один до багатьох" і мала деякі схожості з протоколом передачі пошти.

Оригінальна версія протоколу SMTP підтримувала лише незашифровані 7-бітні текстові повідомлення ASCII без автентифікації. Це призводило до вразливості протоколу перед підробкою, спамом і атакою "людина посередині". Крім того, будь-які двійкові дані потребували кодування у текстовий формат перед передачею. Відсутність належного механізму автентифікації призводила до того, що кожен SMTP-сервер став вразливим ретранслятором електронної пошти.

Сучасний SMTP

RFC 1869 [28] встановив стандарт для розширеного протоколу передачі пошти (ESMTP), який надав загальну структуру для всіх існуючих і майбутніх розширень, спрямованих на додавання додаткових функцій, відсутніх у початковому SMTP.

ESMTP включає узгоджені та контрольовані засоби для ідентифікації клієнтів і серверів ESMTP, а сервери можуть вказувати підтримувані розширення.

Протоколи надсилання повідомлень і SMTP-AUTH, представлені у 1998 і 1999 роках відповідно, відображають нові тенденції в області доставки електронної пошти. Спочатку сервери SMTP переважно використовувалися внутрішньо в організаціях, приймали пошту для організації ззовні і пересилали повідомлення з організації на зовнішні сервери. Проте з часом сервери SMTP (агенти передачі пошти) розширили свою роль, тепер вони можуть ретранслювати пошту ззовні організації. Наприклад, керівник компанії може надсилати електронну пошту під час поїздки через корпоративний сервер SMTP. Цей швидкий розвиток та популярність Інтернету призвели до необхідності включення спеціальних правил та методів ретрансляції пошти в протокол SMTP, а також впровадження механізму автентифікації користувачів для запобігання зловживанням, таким як ретрансляція небажаної електронної пошти (спаму).

Початковий протокол SMTP був базований виключно на тексті ASCII і погано справлявся з передачею двійкових файлів або символів, що належать до неанглійських мов. Для кодування двійкових файлів та їх передачі через SMTP були розроблені стандарти, такі як Multipurpose Internet Mail Extensions (MIME). Сьогоднішні агенти передачі пошти, як правило, підтримують розширення 8BITMIME, що дозволяє передавати деякі двійкові файли майже так само просто, як і звичайний текст (для більшості нетекстових файлів необхідне кодування MIME даних). У 2012 році було розроблено розширення SMTPUTF8 для підтримки тексту UTF-8, що дозволяє використовувати міжнародний вміст та адреси, що містять символи з різних нелатинських алфавітів, наприклад кирилицю або китайські знаки.

Компоненти SMTP

1. Mail User Agent (MUA): це комп’ютерна програма, яка допомагає вам надсилати та отримувати пошту. Він відповідає за створення повідомлень електронної пошти для передачі до агента передачі пошти (MTA).

2. Mail Submission Agent (MSA): це комп'ютерна програма, яка в основному отримує пошту від Mail User Agent (MUA) і взаємодіє з Mail Transfer Agent (MTA) для передачі пошти.

3. Mail Transfer Agent (MTA): це в основному програмне забезпечення, яке виконує роботу з передачі пошти з однієї системи в іншу за допомогою SMTP.

4. Mail Delivery Agent (MDA): Агент доставки пошти або локальний агент доставки — це, по суті, система, яка допомагає доставляти пошту в локальну систему.

Електронна пошта надсилається поштовим клієнтом (агентом користувача електронної пошти, MUA) на поштовий сервер (агентом подання електронної пошти, MSA) за допомогою SMTP на TCP-порт 587(Рис. 2).

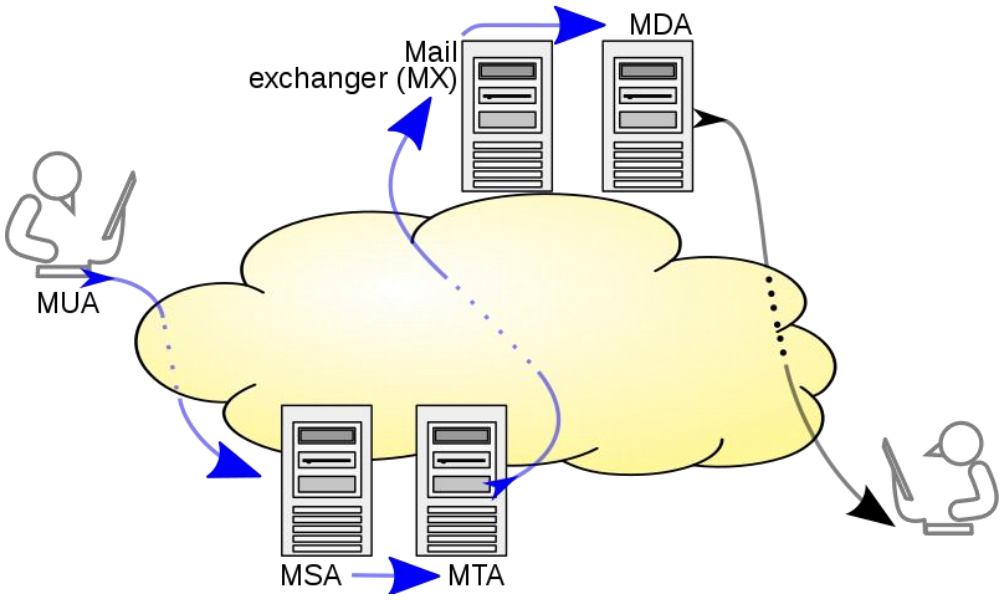

Рисунок 2.2 – Модель обробки пошти SMTP.

Більшість постачальників поштових скриньок все ще дозволяють надсилати на традиційний порт 25. MSA доставляє пошту на свій агент передачі пошти (агент передачі пошти, MTA). Локальну обробку можна виконувати на одній машині або розділити на декілька машин; процеси поштового агента на одній машині можуть обмінюватися файлами, але якщо обробка здійснюється на кількох машинах, вони передають повідомлення один одному за допомогою SMTP, де кожна машина налаштована на використання наступної машини як розумного хоста. Кожен процес є власним MTA (сервером SMTP).

Граничний MTA використовує DNS для пошуку запису MX (обмінник поштою) для домену одержувача (частина електронної адреси праворуч від @). Запис MX містить назву цільового MTA. На основі цільового хоста та інших факторів MTA-відправник вибирає сервер одержувача та підключається до нього для завершення обміну поштою. Передача повідомлень може відбуватися в одному з'єднанні між двома MTA або в серії переходів через системи-посередники. Приймаючий SMTP-сервер може бути кінцевим пунктом призначення, проміжним «ретранслятором» (тобто він зберігає та пересилає повідомлення) або «шлюзом» (тобто він може пересилати повідомлення за допомогою протоколу, відмінного від SMTP). Кожен стрибок є офіційною передачею відповідальності за повідомлення, за допомогою чого сервер-одержувач повинен або доставити повідомлення, або належним чином повідомити про невдачу.

Коли останній перехідний вузол отримує вхідне повідомлення, воно передається агенту доставки пошти (MDA) для локальної доставки. MDA зберігає повідомлення у відповідному форматі поштової скриньки. Аналогічно до процесу надсилання, цей отримуючий етап може здійснюватися за допомогою одного або кількох комп'ютерів, проте на діаграмі вище MDA зображено як окремий ящик поруч з поштовим обмінником. MDA може доставляти повідомлення безпосередньо до сховища або пересилати їх через мережу за допомогою протоколу SMTP або іншого протоколу, наприклад, протоколу локальної передачі пошти (LMTP), що базується на SMTP та використовується для цих цілей.

Після доставки на локальний поштовий сервер пошта зберігається для пакетного отримання аутентифікованими поштовими клієнтами (MUA). Програми кінцевих користувачів, відомі як клієнти електронної пошти, отримують пошту за допомогою протоколу доступу до повідомлень в Інтернеті (IMAP) або протоколу поштового відділення (POP). Протокол IMAP спрощує доступ до пошти та управління збереженими повідомленнями, тоді як протокол POP, який зазвичай використовується для традиційної пошти у форматі mbox або спеціалізованих систем, таких як Microsoft Exchange/Outlook або Lotus Notes/Domino. Клієнти веб-пошти

можуть використовувати будь-який з цих методів, проте протокол отримання часто не є офіційним стандартом.

SMTP визначає механізм транспортування повідомлень, а не їх зміст. Це означає, що він визначає поштовий конверт зі своїми характеристиками, наприклад, відправником конверта, але не включає заголовок та тіло самого повідомлення.

Також у сучасному SMTP використовується два основні протокли шифрування. SSL (Secure Sockets Layer) і TLS (Transport Layer Security) є протоколами шару захисту транспортного рівня, які забезпечують безпеку і конфіденційність під час передачі даних між клієнтом і сервером. Обидва протоколи здатні працювати з протоколом SMTP (Simple Mail Transfer Protocol) для безпечної взаємодії з поштовим сервером [29].

SSL і TLS використовують криптографічні методи для захисту даних від несанкціонованого доступу, перехоплення та модифікації під час трансляції через мережу. Основні принципи роботи SSL / TLS включають наступні етапи:

- Початок з'єднання: Клієнт встановлює з'єднання з сервером SMTP за допомогою протоколу TCP (Transmission Control Protocol). Зазвичай це виконується на порті 25 для SMTP.
- Привітання (Handshake): Клієнт починає привітання, відправляючи запит на з'єднання до сервера SMTP. У відповідь сервер надсилає свій сертифікат, який містить публічний ключ сервера та інші реквізити.
- Перевірка сертифіката: Клієнт перевіряє валідність серверного сертифіката. Валідність включає перевірку цифрового підпису, довіреності видачі сертифіката та терміну його дії. Якщо сертифікат прийнятий, клієнт генерує випадковий симетричний ключ сеансу.
- Обмін ключами: Клієнт зашифровує симетричний ключ сеансу за допомогою публічного ключа сервера з сертифіката та відправляє його серверу. Тепер і клієнт, і сервер знають симетричний ключ для шифрування і розшифрування даних.
- Захищений обмін даними: Клієнт і сервер використовують симетричний ключ сеансу для шифрування і розшифрування даних, які передаються між ними. Це

забезпечує конфіденційність та цілісність даних, оскільки тільки клієнт і сервер мають доступ до симетричного ключа.

- Завершення з'єднання: Після успішного захищеного обміну даними, клієнт і сервер можуть продовжувати комунікацію через захищене з'єднання. Зазвичай клієнт відправляє команду завершення (QUIT) для коректного закриття з'єднання.

SSL і TLS використовують різні версії протоколу і механізми шифрування. Однак термін "SSL" часто використовується загальною формою для вказівки безпечного з'єднання, включаючи і TLS.

За допомогою SSL або TLS можна забезпечити захищене з'єднання з SMTP-сервером, що дозволяє надсилати електронну пошту зашифрованою і захищеною від несанкціонованого доступу. Клієнт і сервер використовують криптографічні методи для шифрування та розшифрування даних, забезпечуючи конфіденційність та безпеку під час передачі інформації.

2.3.4 Можливості безпечної автентифікації та відправки файлів по електронній пошті

Для безпечної аутентифікації при відправці повідомлень через smtp сервер у багатьох розвинутих сервісах використовуюється SASL XOAUTH2.

SASL XOAUTH2 (Simple Authentication and Security Layer, Extensible Authentication Protocol version 2) [30] є механізмом аутентифікації, який використовується для безпечної аутентифікації з використанням токенів доступу (access tokens). Цей механізм дозволяє додаткам отримати доступ до ресурсів користувача без передачі його пароля. SASL XOAUTH2 використовується для аутентифікації з SMTP (Simple Mail Transfer Protocol) під час надсилання електронної пошти через поштовий сервер. Замість передачі пароля користувача, використовується спеціальний токен доступу, який отримується від провайдера ідентифікації, такого як Google або Microsoft.

Основні кроки використання SASL XOAUTH2 з SMTP:

- Отримання токену доступу: Додаток повинен отримати токен доступу від провайдера ідентифікації, використовуючи свої клієнтські ідентифікатори та секрети. Це може включати процес обміну кодом авторизації або використання інших механізмів, які встановлюють ідентичність додатка.
- Формування запиту аутентифікації: Під час надсилання поштового повідомлення через SMTP, додаток формує запит аутентифікації з SASL XOAUTH2. Запит містить інформацію про клієнтські ідентифікатори, токен доступу та інші параметри.
- Відправлення запиту аутентифікації: Запит аутентифікації SASL XOAUTH2 надсилається разом із звичайними SMTP командами через зашифроване з'єднання з SMTP-сервером.
- Перевірка аутентифікації: SMTP-сервер перевіряє запит аутентифікації, розшифровує токен доступу та перевіряє його валідність. Якщо аутентифікація успішна, додаток отримує дозвіл на надсилання повідомлення від імені користувача.
- Надсилання повідомлення: Після успішної аутентифікації, додаток може надіслати електронне повідомлення через SMTP-сервер, використовуючи звичайні SMTP команди.

SASL XOAUTH2 забезпечує безпеку аутентифікації, оскільки пароль користувача не передається через мережу. Замість цього, використовується токен доступу, який має обмежений термін дії і може бути відкликаний користувачем. Це покращує безпеку процесу надсилання електронної пошти та зменшує ризик витоку паролів та робить процес аутентифікації безпечнішим та зручнішим для користувачів.

2.3.5 Огляд стандарту розширення можливостей передачі даних електронною поштою MIME

Особливо важливо відзначити стандарт MIME, який був згаданий декілька разів раніше. Multipurpose Internet Mail Extensions (MIME) - це стандарт Інтернету, який розширює формат електронної пошти, щоб підтримувати текст у різних

символьних наборах, включаючи ASCII, а також дозволяє вкладати аудіо, відео, зображення та прикладні програми. Тіла повідомлення можуть складатися з кількох частин, а інформація в заголовку може бути вказана у символьних наборах, відмінних від ASCII. Повідомлення з використанням формату MIME зазвичай передаються за допомогою стандартних протоколів, таких як SMTP, POP і IMAP.

Хоча MIME був розроблений головним чином для SMTP, його типи вмісту також використовуються в інших протоколах зв'язку. У протоколі HTTP для Всесвітньої павутини сервери включають заголовок MIME перед будь-яким веб-запитом. Клієнти використовують тип вмісту або заголовок медіа-типу, щоб вибрати відповідну програму для перегляду вказаного типу даних.

У червні 1992 року стандарт MIME визначив набір методів для представлення двійкових даних у форматах, відмінних від ASCII. Поле заголовка MIME "Content-transfer-encoding" має подвійне значення: воно вказує, чи використовувалась схема кодування для перетворення двійкових даних на текст, додатково до оригінального кодування, як вказано в заголовку "Content-Type":

- Якщо була використана схема кодування для перетворення двійкових даних на текст, вказується конкретний метод кодування.
- Якщо схема кодування не використовувалась, вказується описовий маркер для формату вмісту щодо наявності 8-бітового або двійкового вмісту.

RFC і список кодувань передачі IANA визначають нижченаведені значення, які нечутливі до регістру. Терміни "7-бітний", "8-бітний" і "двійковий" вказують на те, що використання кодування двійкових даних до текстового формату не відбувалося додатково до оригінального кодування. У таких випадках поле заголовка фактично не потрібне для клієнта електронної пошти для декодування тіла повідомлення, але воно може бути корисним як індикатор типу надсиланого об'єкта. Значення "quoted-printable" і "base64" повідомляють клієнтові електронної пошти, що було застосовано кодування двійкових даних до тексту, і необхідне початкове декодування, перш ніж повідомлення можна буде прочитати у вихідному кодуванні, наприклад, UTF-8.

Ці значення підходять для використання зі звичайним SMTP:

- "7 біт" - Це значення є значенням за замовчуванням.
- "quoted-printable" - використовується для кодування довільних послідовностей октетів у форму, яка відповідає правилам 7-біт. Цей метод був розроблений, щоб бути ефективним і зрозумілим для людей, особливо при застосуванні до текстових даних, що складаються переважно з символів US-ASCII, але містять невеликий відсоток байтів зі значеннями поза цим діапазоном.
- "base64" - використовується для кодування довільних послідовностей октетів у форму, яка відповідає правилам 7-біт. Цей метод був розроблений для ефективного кодування нетекстових 8-бітних і двійкових даних. Час від часу його також використовують для текстових даних, які часто включають символи, відмінні від американського ASCII.

Висновки до розділу 2

У цьому розділі були розглянуті протоколи які надають можливість передавати мультимедійні файли по мережі Інтернет, які реалізують основні принципи передачі файлів.

Основними протоколами для транспортування файлів є :

FTP (File Transfer Protocol) - це стандартний мережевий протокол, який використовується для передачі файлів між клієнтом і сервером через мережу на основі TCP/IP. Він був розроблений у 1970-х роках і з тих пір став одним із найпоширеніших методів передачі файлів в Інтернеті. FTP працює в архітектурі клієнт-сервер, де клієнт ініціює підключення до сервера та запитує передачу файлів. Сервер, який зазвичай є виділеним FTP-сервером, прослуховує вхідні з’єднання та відповідає на запити клієнтів.

HTTP (Hyper Text Transfer Protocol) - це протокол, який використовується для спілкування у Всесвітній павутині. Це основа передачі даних для Інтернету, уможливлюючи пошук і відображення веб-сторінок та інших ресурсів. HTTP — це протокол прикладного рівня, який працює поверх набору протоколів TCP/IP. HTTP в основному призначений для передачі текстових даних, наприклад документів HTML,

але цим не обмежується. Він також може передавати різні типи файлів, включаючи зображення, відео, аудіофайли тощо.

Окремо розглянули протоколи для відправлення та отримання електронної пошти, як одного із провідних способів обміну інформацією, в тому числі і файлами. POP3 (протокол поштового офісу версії 3), IMAP (протокол доступу до повідомлень в Інтернеті) і SMTP (простий протокол передачі пошти) — це всі протоколи, які використовуються для спілкування електронною поштою. Хоча вони в основному зосереджені на обробці електронної пошти, у них є певні можливості, пов’язані з передачею файлів.

SMTP — це стандартний протокол, який використовується для надсилання повідомлень електронної пошти між серверами електронної пошти. Він відповідає за передачу електронних листів із поштової програми клієнта на сервер електронної пошти одержувача. SMTP безпосередньо не обробляє передачу файлів, крім прикріплення файлів до повідомлень електронної пошти. Під час надсилання електронного листа можна додати вкладені файли, що дозволить одержувачу завантажити та зберегти вкладені файли.

Вкладення в SMTP обробляються шляхом кодування файлів і включення їх як частини повідомлення електронної пошти. Багатоцільові розширення Інтернет-пошти (MIME) - це стандарт Інтернету, який розширює формат електронної пошти, щоб підтримувати текст у різних символьних наборах, включаючи ASCII, а також дозволяє вкладати аудіо, відео, зображення та прикладні програми. MIME використовуються для кодування вкладень у SMTP, та дозволяє надсилати двійкові дані, наприклад файли, як текст у повідомленні електронної пошти. Повідомлення електронної пошти структуровано як багатокомпонентне повідомлення MIME, яке може містити як текст, так і вкладення. Він складається з кількох частин, кожна зі своїм типом вмісту.

РОЗДІЛ 3
ПРОГРАМНІ ЗАСОБИ ОБРОБКИ МЕТАДАНИХ В МУЛЬТИМЕДІЙНИХ ФАЙЛАХ ДЛЯ ЗАХИЩЕНОЇ ПЕРЕДАЧІ

3.1. Визначення вимог для створення програмного модуля

Головним завданням програмного модуля є захист конфіденційності користувача шляхом анонімізації метаданих. Додатковими завданнями є відправка отриманого після обробки мультимедійного файлу захищеним шляхом, а також надання зручного доступу до функцій застосунку. Оскільки метадані можуть містити різноманітну інформацію про користувача, таку як географічне положення, час створення файлу, тип пристрою та авторство, програмний модуль спрямований на зменшення ризиків, пов'язаних з цією інформацією. Він надає користувачам інструменти для ефективного захисту їхньої приватності і можливості безпечної передачі файлу по зашифрованому каналу.

Функціональні вимоги

Перегляд метаданих

Програмний модуль має забезпечувати зручний та швидкий перегляд метаданих мультимедійних файлів користувачами. Це означає, що програма повинна аналізувати файл, виділяти всю доступну інформацію з метаданих та відображати її в зрозумілому форматі. Цей процес включає зчитування даних про автора, дату створення файлу, його розмір, тип пристрою, на якому він був створений, а також, якщо можливо, географічну локацію, де файл був створений.

Анонімізація

Однак головною метою розробленого програмного модуля є не лише виявлення метаданих, але й надання користувачам можливості анонімізувати цю інформацію для захисту їхньої приватності. Це може бути реалізовано шляхом простого видалення чутливих даних з метаданих або видалення всіх метаданих мультимедійного файлу, якщо це допустимо.

Передача файлу

Додатковою вимогою було також реалізація можливості передачі обробленого файлу прямо з застосунку. Для цього можна використати будь-який поштовий сервер SMTP, який би підтримував зашифровану передачу та автентифікацію, а також вкладення MIME. Під такі вимоги підпадають майже всі сучасні SMTP сервери, тож можна обрати найбільш зручний для користувача.

Нефункціональні вимоги

- Забезпечити простоту використання додатку
- Сумісність з Windows та UNIX подібними операціними системами
- Швидкість роботи з файлами великого розміру
- Підтримка широкого ряду різних типів мультимедійних файлів

3.2. Вибір технологій та інструментів розробки

Основною мовою застосунку є мова програмування середнього рівня C, на цій мові реалізоване основне тіло програми яке дозволяє легко проводити операції перегляду та обробки метаданих мультимедійних файлів. Додатково для реалізації захищеної передачі оброблених файлів використовується скрипт написаний на мові Python. У ньому відбувається передача вкладення MIME по протоколу SMTP з використанням SSL шифрування, де вкладенням MIME виступає попередньо оброблений мультимедійний файл.

Для створення власного програмного модуля було доречно використати деякі вже існуючі застосунки які дозволяють здійснювати широкий спектр дій з метаданими мультимедійних файлів, але мають деякі проблеми, пов’язані в основному зі зручністю дій, необхідності дослідження документації та відсутності деяких функцій. Тому я вирішив розробити застосунок на мові C, для широкої сумісності з різними операційними системами. Окрім мови C застосунок також використовує python скрипт який може бути використаний для зручної та зашифрованої передачі мультимедійного файлу електронною поштою після того як його метадані були оброблені. Обробка відбувається з використанням застосунку ExifTool та FFmpeg, для графічних зображень та відеофайлів відповідно. ExifTool і

FFmpeg є широко використовуваними програмами з відкритим кодом для керування метаданими фото- та відеофайлів. ExifTool — це потужний інструмент командного рядка, який дозволяє читати, записувати та маніпулювати метаданими в різних форматах файлів, включаючи фотографії (JPEG, TIFF, RAW).

ExifTool [32] може видобувати, змінювати та видаляти теги метаданих, такі як марка та модель камери, GPS-координати, дата й час, налаштування експозиції тощо. Це може бути корисним для організації, сортування та каталогізації фотографій і відео. ExifTool підтримує масову обробку, що дозволяє застосовувати зміни до кількох файлів одночасно. Це особливо корисно під час роботи з великими колекціями мультимедійних файлів. При цьому ExifTool підтримує широкий спектр форматів файлів, що робить його універсальним для роботи з різними типами медіафайлів. ExifTool надає широкі можливості та гнучкість, дозволяючи вам налаштовувати операції з метаданими відповідно до ваших конкретних вимог.

Що стосується безпеки, ExifTool загалом вважається безпечним у використанні. Це проект із відкритим вихідним кодом із великою базою користувачів, який активно підтримується протягом багатьох років. Однак, як і будь-яке інше програмне забезпечення, рекомендується завантажувати його з надійних джерел (наприклад, з офіційного веб-сайту) і бути обережними під час виконання операцій командного рядка, щоб уникнути небажаних наслідків. Для вирішення цього мій застосунок у разі відсутності ExifTool автоматично направить користувача на офіційний сайт для встановлення свіжої версії ExifTool, а також користувачу не доведеться виконувати операції з командним рядком ExifTool напряму, оскільки мій додаток містить у собі перевірені команди для дій з метаданими мультимедійних файлів.

FFmpeg — це міжплатформний інструмент командного рядка для роботи з мультимедійними файлами, зокрема відео. Хоча він насамперед відомий своїми потужними можливостями кодування та перекодування відео, він також пропонує функції, пов’язані з керуванням метаданими. FFmpeg може витягувати метадані з відеофайлів, зокрема інформацію про відеокодек, аудіокодек, тривалість, роздільну здатність тощо. Це може бути корисним для аналізу та організації вашої бібліотеки відео. FFmpeg також може змінювати та додавати метадані до відеофайлів. Ви можете

вказати такі теги, як назва, виконавець, інформація про авторські права та призначені для користувача метадані. FFmpeg підтримує широкий спектр відео- та аудіоформатів, що дозволяє конвертувати файли, зберігаючи або змінюючи метадані під час процесу. Подібно до ExifTool, FFmpeg є проектом з відкритим кодом із великою спільнотою користувачів і розробників, що робить його загалом безпечним у використанні. Однак завжди рекомендується завантажувати FFmpeg із надійних джерел і бути обережними під час використання інструментів командного рядка, щоб уникнути потенційних ризиків або небажаних наслідків. Для уникнення цих недоліків мій застосунок реалізує дії аналогічні до тих що виконуються по відношенню до ExifTool.

Таким чином, і ExifTool, і FFmpeg є цінними програмами з відкритим вихідним кодом для керування метаданими у фото- та відеофайлах. Вони пропонують широкі функціональні можливості, можливості пакетної обробки та підтримку різних форматів файлів.

Загальна структура взаємодії програмних модулів, які реалізують функції програмного застосунку по видаленню метаданих з мультимедійних файлів представлена на рисунку 3.1.

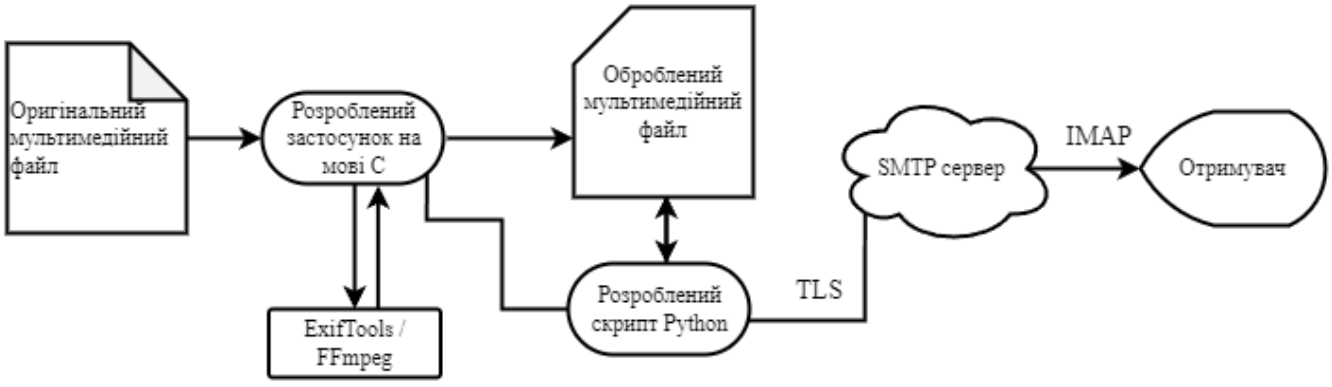

Рисунок 3.1 - Структура взаємодії модулів програми.

Основне тіло застосунку розроблене на мові C. Мова програмування C це мова комп'ютерного програмування загального призначення. Він був створений у 1970-х роках Деннісом Річі та залишається дуже широко використовуваним і впливовим. За дизайном функції C чітко відображають можливості цільових ЦП. Він знайшов

тривале використання в операційних системах, драйверах пристроїв, стеках протоколів. С зазвичай використовується в комп’ютерних архітектурах, які варіюються від найбільших суперкомп’ютерів до найменших мікроконтролерів і вбудованих систем. С має високу продуктивність, низький рівень абстракції та прямий доступ до апаратного забезпечення. Це особливо важливо при розробці застосунків, які мають обробляти великі обсяги даних, як мультимедійні файли. Мова С дозволяє ефективно управляти ресурсами та оптимізувати роботу з пам'яттю, що може бути корисним для забезпечення швидкості та ефективності обробки даних.

Додаткова частина програми розроблена на мові Python. Python - це високорівнева об'єктно-орієнтована мова програмування з інтерпретатором, яка має динамічну семантику. Його вбудовані структури даних в поєднанні з динамічною типізацією та динамічним зв'язуванням роблять його надзвичайно привабливим для швидкого розроблення додатків. Крім того, він часто використовується як мова сценаріїв або мова з'єднувача для інтеграції існуючих компонентів.

У розробленому застосунку скрипт Python дозволяє користувачу надіслати очищений мультимедійний файл електронною поштою прямо з додатку. Його авторизаційні дані будуть захищені через Simple Authentication and Security Layer, з використанням XOAUTH2 , а дані що передаються шифруванням SSL. Це дозволить безпечно та зручно надіслати повідомлення разом з мультимедійними файлами.
Для реалізації використовується декілька бібліотек.

Python socket:

Бібліотека socket в Python надає інтерфейс для роботи з мережевими сокетами. Вона дозволяє створювати клієнтські та серверні додатки, які можуть взаємодіяти через різні мережеві протоколи, такі як TCP або UDP. Socket дозволяє передавати дані через мережу, слухати порти, встановлювати з'єднання та обмінюватися повідомленнями між різними вузлами в мережі.

Python ssl:

Бібліотека ssl в Python надає функціональність для роботи з шифруванням та захистом з'єднань по мережі. Вона дозволяє створювати захищені TLS/SSL з'єднання

між клієнтом та сервером. Бібліотека ssl забезпечує можливості для автентифікації, шифрування та перевірки цілісності даних, що передаються через мережу.

Python base64:

Бібліотека base64 в Python надає інструменти для кодування та декодування даних в форматі Base64. Base64 - це стандартний спосіб перетворення бінарних даних в текстовий формат, що складається з печатних символів ASCII. Base64 часто використовується для представлення бінарних даних у вигляді тексту, наприклад, при передачі файлів через протоколи електронної пошти або в мережевих протоколах.

Python email:

Бібліотека email в Python надає функціональність для створення, відправлення та обробки електронних листів. Вона дозволяє формувати електронні листи з заголовками, текстом, вкладеннями та іншими додатковими властивостями. Бібліотека email також забезпечує інструменти для розбору та аналізу отриманих листів, що дозволяє автоматизувати обробку електронної пошти в програмах на Python. В бібліотеці Python email, модуль email.mime.multipart надає можливості для створення та обробки MIME-повідомлень з багатьма частинами. Клас MIMEMultipart з модулю email.mime.multipart є контейнером для кількох частин повідомлення. Він дозволяє створювати повідомлення з багатьма під-частинами, які можуть мати різний тип контенту і вкладені файли.

3.3. Реалізація основних функціональних компонентів

Розглянемо детально деякі основні функціональні компоненти, реалізовані у основному тілі додатку та у модулі передачі трафіку. При запуску додатку нас зустрічає меню, в якому потрібно визначити тип мультимедійного файлу – фото чи відео. Після чого користувач може ввести ім’я файлу самостійно, або ж використати * щоб обрати усі мультимедійні файли у папці.

Для цього використовується функція пошуку search().

Основні кроки, які виконує функція search(), такі:

- Відкривається файл з назвою, збереженою в змінній file, за допомогою функції fopen(). Вказується режим "r" (тобто для читання файлу).
- Перевіряється, чи вдалося відкрити файл. Якщо файл не знайдено (тобто f має значення NULL), виводиться повідомлення про помилку "ERROR: FILE NOT FOUND!" за допомогою функції printf().
- Файл закривається за допомогою функції fclose(). Важливо звільнити ресурси після закінчення роботи з файлом.

Далі відбувається демонстрація існуючих метаданих та відбувається їх очистка.

Для цього використовується функція cleanin()

У випадку, якщо змінна visit дорівнює 0 (тобто обробка метаданих ще не виконувалася), виконується наступне:

- У рядок eff копіюється команда "exiftool ".
- До рядка eff додається назва файлу (file).
- Виконується команда system(extra), яка запускає виконання команди в системному шелі.
- До рядка eff додається " -all=".
- Змінна check отримує значення, що повертається від system(eff).

Ця функція використовує зовнішню утиліту "exiftool" для виконання очищення метаданих файлу. Вона перевіряє розширення файлу та виконує відповідні команди залежно від умов. Після чого проводиться порвіняння кожного обробленого файлу для визначення об’єму проведених робот. За це відповідає функція compare() Деталі можна переглянути у файлах логування, названі input та output які будуть створені після успішного завершення очищення.

Подальша відправка оброблених файлів буде виконана через python скрипт. Спершу застосунок встановлює з'єднання з SMTP-сервером, аутентифікується і передає облікові дані (логін та пароль) для доступу до поштової скриньки на сервері.

- Встановлюється значення змінної mailserver як "smtp.gmail.com".
- Створюється сокет (clientSocket) та встановлюється TCP-з'єднання з сервером за допомогою функції connect(). З'єднання також обгортається SSL за допомогою ssl.wrap_socket(), щоб забезпечити безпечний обмін даними.

- Отримується відповідь (recv) від сервера після підключення. Це повідомлення з кодом 220, що підтверджує успішне встановлення з'єднання. Якщо отриманий код не дорівнює "22", виводиться повідомлення про невдале отримання коду 220.
- Відправляється команда "EHLO Vitaliy" на сервер за допомогою send(). Отримується відповідь (recv1) від сервера, яка містить підтвердження про підтримку протоколу і список доступних функцій.
- Задаються значення змінних Username і Password, які містять облікові дані (логін та пароль) для аутентифікації на сервері.
- Логін (Username) перекодовується у формат Base64 за допомогою base64.b64encode(). Результат зберігається у змінній UsrCodedString.
- Відправляється команда AUTH LOGIN разом із закодованим логіном (Auth) на сервер за допомогою send(). Отримується відповідь (recvFromAuth) від сервера щодо статусу аутентифікації.
- Пароль (Password) перекодовується у формат Base64, а результат додається до команди AUTH LOGIN для відправки на сервер. Отримується відповідь (recvFromAuth) від сервера щодо статусу аутентифікації.

Відправка мультимедійного файлу, як вкладення електронної пошти відправляється наступним чином :

- Створюється об'єкт msg класу MIMEMultipart(), який представляє собою багаточастинне повідомлення (MIME multipart message).
- Виконується додавання текстового повідомлення до об'єкту msg за допомогою методу attach(). Текст повідомлення передається через параметр MIMEText().
- Перевіряється, чи потрібно додати файл до повідомлення. Якщо змінна needfile дорівнює 'y', виконується наступне:
- Відкривається файл (filename) у режимі читання бінарного режиму ("rb").
- Створюється об'єкт part класу MIMEBase з типом application та octet-stream.

- Встановлюється вміст файлу, прочитаний з attachment, як вміст part за допомогою методу set_payload().
- Кодується вміст part в Base64 за допомогою методу encode_base64() з модуля encoders.
- Додається заголовок Content-Disposition до part, який вказує назву файлу.
- Об'єкт part додається до об'єкту msg за допомогою методу attach().
- Змінна msg перетворюється в рядок (compl_msg) за допомогою методу as_string().
- Виконується надсилання повідомлення:
- Рядок compl_msg кодується у байти (encode()) та відправляється через clientSocket за допомогою методу send().
- Рядок endmsg кодується у байти (encode()) та відправляється через clientSocket за допомогою методу send().
- Виконується команда "QUIT" для завершення сеансу з SMTP-сервером. Рядок "QUIT\r\n" кодується у байти (encode()) та відправляється через clientSocket за допомогою методу send().

Після чого отримувач матиме змогу отримати повний мультимедійний файл в оригінальній якості на своєму поштовому клієнті, але будь-які конфіденційні дані які містилися у метаданих цього мультимедійного файлу будуть видалені, що призведе до забезпечення конфіденційності користувача який скористався розробленим програмним застосунком.

Висновок до розділу 3

У даному розділі було зроблено огляд архітектури програмного модуля, використані технології та інструменти під час розробки, а також надано опис основних функцій додатку, їх призначення та робочий принцип. Крім того, були визначені функціональні та нефункціональні вимоги, що мали бути враховані під час створення програми.

Програмний модуль був реалізований на мові програмування C з використанням базових бібліотек. Цей додаток дозволяє користувачу зручно видаляти та переглядати метадані файлів, а також передавати ці файли по зашифрованим каналам, використовуючи електронну пошту.

ВИСНОВОК

У кваліфікаційній роботі було розроблено комплекс програмних засобів для захисту конфіденційності користувача шляхом видалення метеданих мультимедійних файлів при передачі мультимедіа трафіку, з використанням безпечних протоколів авторизації та передачі даних SSL по протоколу SMTP.

У першому розділі кваліфікаційної роботи було здійснено аналіз стандартів метаданих мультимедійних файлів та законодавства, що регулює збереження інформації в метаданих. Була розглянута класифікація метаданих відповідно до розглянутих стандартів, включаючи технічні, описові та адміністративні метадані. Крім того, було описано основні стандарти метаданих, що використовуються у мультимедійних файлах. Також були розглянуті потенційні загрози конфіденційності, пов'язані з метаданими мультимедійних файлів, зокрема розкриття геолокації, особистої інформації та технічних характеристик пристроїв. Було проведено огляд сучасних методів та технік обробки метаданих, які спрямовані на запобігання цим загрозам.

У другій частині кваліфікаційної роботи були розглянуті різні типи протоколів передачі файлів в мережі Інтернет (FTP, HTTP, SMTP). Проведений аналіз основних принципів роботи цих протоколів, а також їх сучасні покращення. Описаний прицнип передачі файлів через електронну пошту(IMAP POP3 SMTP), та розглянуті принципи захисту конфіденційності при відправці мультимедійних вкладень. Для цього детально проаналізовано простий протокол передачі пошти - SMTP та його модулі, включаючи протокол простої аутентифікації та рівня безпеки SASL XOAUTH2, мультизадачне розширення інтернет пошти MIME і два сучасні типи шифрування SSL – рівень захищених сокетів та TLS – безпека транспортного рівня.

У третій частині кваліфікаційної роботи на основі проаналізованих у першій та другій частині даних було розроблено програмний застосунок який дозволяє користувачу зручно видаляти та переглядати метадані файлів, а також передавати ці файли по зашифрованим каналам, використовуючи електронну пошту.

Виходячи із поставленої мети кваліфікаційної роботи були виконані наступні завдання:

- Досліджено класифікацію та стандарти метаданих мультимедійних файлів. та можливих загроз конфіденційності при передачі метаданих у пов’язаних з ними мультимедійних ресурсах.
- Проведено аналіз протоколів передачі мультимедійних даних різних типів по стандартним протоколам зв’язку в мережі Інтернет, визначені можливості забезпечення більш безпечної передачі мультимедійних ресурсів використовуючи існуючі засоби.
- Реалізовано програмний застосунок який дозволяє користувачу зручно видаляти та переглядати метадані файлів, а також передавати ці файли по зашифрованим каналам, використовуючи електронну пошту.

Всі завдання було виконано в повному обсязі.

СПИСОК ДЖЕРЕЛ

1. Lee E. 2021 worldwide image capture forecast: 2020 – 2025 - rise above research [Електронний ресурс] / Ed Lee // Rise Above Research. – Режим доступу: https://riseaboveresearch.com/rar-reports/2021-worldwide-image-capture-forecast-2020-2025/

2. Ignorance and distrust prevail about what companies and governments do with personal data [Електронний ресурс] / уклад. Ipsos. – DAVOS/PARIS : Reuters, 2019. – 2 с. – Режим доступу: https://www.ipsos.com/en/ignorance-and-distrust-prevail-about-what-companies-and-governments-do-personal-data

3. Quach S. Digital technologies: tensions in privacy and data - Journal of the Academy of Marketing Science [Електронний ресурс] / Sara Quach, Park Thaichon // SpringerLink. – Режим доступу: https://link.springer.com/article/10.1007/s11747-022-00845-y

4. Відео про мінування мосту на шляху евакуації біженців з ОРДЛО зняли у 2019 році [Електронний ресурс] / Редакція ФОКУС. – Київ : Фокус, 2022. – Режим доступу: https://focus.ua/uk/voennye-novosti/507170-video-o-minirovanii-mosta-na-puti-evakuacii-bezhencev-iz-ordlo-bylo-snyato-v-2019-godu

5. Basic Metadata: Don't Process Without It Adding Contact and Copyright Metadata to Your RAW Processing Workflow by Ethan G. Salwen, AfterCapture magazine, Oct/Nov 2007 [Електронний ресурс] Режим доступу : https://www.rangefinderonline.com/repository/rf/articles/pdf/Mar08_48.pdf

6. JEITA Exchangeable image file format for digital still cameras: Exif Version 2.2 JEITA CP-3451, established April 2002. Published by Japan Electronics and Information Technology Industries Association (JEITA) English translation of the standard, though the Japanese original is authoritative.

http://it.jeita.or.jp/document/publica/standard/exif/english/Exife.pdf

7. The IPTC-NAA standards [Електронний ресурс] – Режим доступу: https://controlledvocabulary.com/imagedatabases/iptc_naa.html.

8. IIM - IPTC [Електронний ресурс] // IPTC. – Режим доступу: https://www.iptc.org/standards/iim/

9. Standards: IPTC core & extensions | photometadata.org [Електронний ресурс] // Are you meta-smart? | Photometadata.org. – Режим доступу: https://photometadata.org/META-Resources-metadata-types-standards-IPTC-Core-and-extensions?subject=IPTC%20Core.

10. Plus [Електронний ресурс] // :: PLUS ::. – Режим доступу: https://www.useplus.com/.

11. XMP metadata [Електронний ресурс] // Adobe Help Center. – Режим доступу: https://helpx.adobe.com/after-effects/using/xmp-metadata.html.

12. What is an EXIF file? [Електронний ресурс]. – Режим доступу: https://www.adobe.com/creativecloud/file-types/image/raster/exif-file.html.

13. Dublin core [Електронний ресурс]. – Режим доступу: https://www.dublincore.org.

14. Про захист персональних даних [Електронний ресурс] : Закон України від 01.06.2010 р. № 2297-VI : станом на 27 жовт. 2022 р. – Режим доступу: https://zakon.rada.gov.ua/laws/show/2297-17#Text

15. Про інформацію [Електронний ресурс] : Закон України від 02.10.1992 р. № 2657-XII : станом на 31 берез. 2023 р. – Режим доступу: https://zakon.rada.gov.ua/laws/show/2657-12#Text.

16. General data protection regulation (GDPR) – official legal text [Електронний ресурс] // General Data Protection Regulation (GDPR). – Режим доступу: https://gdpr-info.eu/.

17. Ivan Parkhomenko, Vitalii Kostiuchenko, Pavlo Horbatiuk. The Crucial Role of Securing Digital Image Metadata: Protecting Privacy, Authenticity, and Integrity. VI Міжнародна науково-практична конференція “Проблеми кібербезпеки інформаційно-телекомунікаційних систем” (PCSITS), Kyiv - P. 59-60.

18. Face/Off: preventing privacy leakage from photos in social networks [Електронний ресурс] / Elias Athanasopoulos [та ін.]. – 2015. – Режим доступу: https://www.researchgate.net/publication/301419612_FaceOff_Preventing_Privacy_Leakage_From_Photos_in_Social_Networks.

19. Menfors M. Geotagging in social media : exploring the privacy paradox [Електронний ресурс] : thesis / Menfors Martina, Fernstedt Felicia. – [Б. м.], 2015. – Режим доступу: https://www.diva-portal.org/smash/get/diva2:896762/FULLTEXT.pdf

20. Anonymisation: managing data protection risk code of practice [Електронний ресурс]. – 2012. – Режим доступу: https://ico.org.uk/media/1061/anonymisation-code.

21. Агенція Європейського Союзу з питань основоположних прав та Рада Європи Посібник з європейського права у сфері захисту персональних даних [Електронний ресурс] / Агенція Європейського Союзу з питань основоположних прав та Рада Європи, 2018. – 436 с. – Режим доступу: https://www.echr.coe.int/Documents/Handbook_data_protection_UKR

22. RFC 959 [Електронний ресурс]. – На заміну RFC 765. – FILE TRANSFER PROTOCOL (FTP) – [Б. м. : б. в.], 1985. – Режим доступу: https://datatracker.ietf.org/doc/html/rfc959.

23. RFC 2068 [Електронний ресурс]. – Hypertext Transfer Protocol -- HTTP/1.1– [Б. м. : б. в.], 1997. – Режим доступу: https://datatracker.ietf.org/doc/html/rfc2068.

24. Awati R. What is POP3 (Post Office Protocol 3)? [Електронний ресурс] / Rahul Awati. – Режим доступу: https://www.techtarget.com/whatis/definition/POP3-Post Office-Protocol 3#:~:text=POP3%20is%20a%20oneway,from%20the%20server%20using%20POP3.

25. RFC 9051 [Електронний ресурс]. – Internet Message Access Protocol (IMAP)– [Б. м. : б. в.], 2021. – Режим доступу: https://datatracker.ietf.org/doc/html/rfc9051.

26. RFC 8314 [Електронний ресурс]. – Use of Transport Layer Security (TLS) for Email Submission and Access – [Б. м. : б. в.], 2018. – Режим доступу: https://datatracker.ietf.org/doc/html/rfc8314.

27. RFC 788 [Електронний ресурс]. – SIMPLE MAIL TRANSFER PROTOCOL– [Б. м. : б. в.], 1981. – Режим доступу: https://datatracker.ietf.org/doc/html/rfc1981.

28. RFC 1869 [Електронний ресурс]. – SMTP Service Extensions. – [Б. м. : б. в.], 1995. – Режим доступу: https://datatracker.ietf.org/doc/html/rfc1869.

29 RFC 4422 [Електронний ресурс]. –Simple Authentication and Security Layer (SASL) – [Б. м. : б. в.], 2013. – Режим доступу: https://datatracker.ietf.org/doc/html/rfc4422.

30. RFC 6838 [Електронний ресурс]. – Media Type Specifications and Registration Procedures– [Б. м. : б. в.], 2013. – Режим доступу: https://datatracker.ietf.org/doc/html/rfc6838.

32. Blackden C. Everything You Ever Wanted to Know about ExifTool [Електронний ресурс] / Chris Blackden // ATA Learning. – Режим доступу: https://adamtheautomator.com/exiftool/.

ДОДАТОК А
КОД ЗАСТОСУНКУ ДЛЯ ОБРОБКИ МЕТАДАНИХ

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <stdbool.h>
#include <sys/stat.h>

#define MAX_BUFF_SIZE 200
#define SANITIZE_COMMAND_TEMPLATE "exiftool %s -all="
#define INPUT_LOG_COMMAND_TEMPLATE "exiftool %s > input.txt"
#define VIDEO_LOG_COMMAND_TEMPLATE "exiftool %s > video_input_metadata.txt"

char file[MAX_BUFF_SIZE], fil[50], vfile[MAX_BUFF_SIZE], vfil[50];



void pause()
{
   printf("\n");
   system("python smtp.py");
   printf("\n");
}

void checker()
{
   int check;
   char command[MAX_BUFF_SIZE];



   snprintf(command, sizeof(command), COMMAND_TEMPLATE, file);

   check = system(command);
   if(check != 0)
   {
      printf("\n\nERROR Output.txt could not be created!\n");
```

```
        pause();
        exit(EXIT_FAILURE);
    }
}
void vchecker()
{
    int check;
    char command[MAX_BUFF_SIZE];


    snprintf(command, sizeof(command), VIDEO_COMMAND_TEMPLATE, vfil);

    check = system(command);
    if(check != 0)
    {
        printf("\n\nERROR Output.txt could not be created!");
        pause();
        exit(EXIT_FAILURE);
    }
}

void input()
{
    system("cls");
    printf("\n\t|----Metadata Tools----|\n");
    printf("\n\n Enter Image name:");
    scanf("%49s", fil);

    snprintf(file, sizeof(file), IMAGE_COMMAND_TEMPLATE, fil);
}



bool endswith(const char* str1, const char* str2)
{
    size_t str1_len = strlen(str1);
    size_t str2_len = strlen(str2);

    if (str1_len >= str2_len) {
```

```
        return strcasecmp(str1 + str1_len - str2_len, str2) == 0;
    }

    return false;
}


void cleanin()
{
    int check;
    char command[MAX_BUFF_SIZE];

    if(endswith(file,"tiff") || endswith(file,"tif"))
    {
        snprintf(command, sizeof(command), "exiftool -all= -CommonIFD0= %s", file);
        check = system(command);
    }
    else
    {
        snprintf(command, sizeof(command), "exiftool %s -s -canon ", file);
        printf("%s", command);
        system(command);

        snprintf(command, sizeof(command), SANITIZE_COMMAND_TEMPLATE, file);
        check = system(command);
    }

    if(check != 0)
    {
        printf("\n\nERROR Image Cleaning failed");
        pause();
        exit(EXIT_FAILURE);
    }
}

void see()
{
            int check;
```

```
    char command[MAX_BUFF_SIZE];
          snprintf(command, sizeof(command), "exiftool %s -G1 -a -s", file);
        printf("%s", command);
        system(command);

}

void ichecker()
{
    int check;
    char command[MAX_BUFF_SIZE];

    snprintf(command, sizeof(command), INPUT_LOG_COMMAND_TEMPLATE, file);

    check = system(command);
    if(check != 0)
    {
        printf("\n\nERROR Input log could not be created ");
        pause();
        exit(EXIT_FAILURE);
    }
}



void ivchecker()
{
    int check;
    char command[MAX_BUFF_SIZE];

    snprintf(command, sizeof(command), VIDEO_LOG_COMMAND_TEMPLATE, vfil);

    check = system(command);
    if(check != 0)
    {
        printf("\n\nERROR Input log could not be created");
        pause();
        exit(EXIT_FAILURE);
    }
```

```
}


void search()
{
   FILE *f;

   if(strstr(file, "*.") == NULL){
      f=fopen(file,"r");
      if(f == NULL)
      {
         printf("\n\nERROR: FILE NOT FOUND!\n\n");
         pause();
         exit(EXIT_FAILURE);
      }
      fclose(f);
   }
}


long get_file_size(char *filename)
{
   struct stat st;

   if(stat(filename, &st) == 0)
      return st.st_size;

   return -1;
}

void compare()
{
   long size_input = get_file_size("input.txt");
   long size_output = get_file_size("output.txt");
   long size_file_original = get_file_size(file);
   char file_original[MAX_BUFF_SIZE];

   snprintf(file_original, sizeof(file_original), "%s_original", file);
```

```
    long size_file = get_file_size(file_original);

    if(size_input == size_output)
    {
        printf("\n\nFile is already cleaned!\n");
    }
    else if(size_input > size_output)
    {
        printf("\n\nMetadata Cleaned Successfully!\n");
        printf("Size before cleaning: %ld bytes, \n", size_file_original);
        printf("Size after cleaning: %ld bytes.\n", size_file);
        printf("Total loss : %ld bytes.", size_file_original - size_file);
    }
    else
    {
        printf("\n\nCleaning done with errors!\n");
    }
}

void vinput()
{
    system("cls");
    printf("\n\t |----- Video sanitisation tool -----|\n");
    printf("\n\n Enter Video name:");
    scanf("%30s",vfil);
    snprintf(vfile, sizeof(vfile), "Videos\\%s", vfil);
}

void run() .
{
    input();
    search();
    ichecker();
    sanitize();
    checker();
    compare();
    pause();
}
```

```
void vdetect(char *filename)
{
   if(access(filename, F_OK) != 0)
   {
      printf("\n\nERROR: FILE NOT FOUND!\n\n");
      pause();
      exit(EXIT_FAILURE);
   }
}

void vcleaning()
{

   int status = 0;
   char buffer[MAX_BUFF_SIZE];
   char fvfile[MAX_BUFF_SIZE];

   snprintf(fvfile, sizeof(fvfile), "Videos\\final_%s", vfil);
   snprintf(buffer, sizeof(buffer), "ffmpeg -i %s -map_metadata -1 -c:v copy -c:a copy %s", vfile, fvfile);

   status = system(buffer);
   if(status != 0)
   {
      printf("\n\n :(  <---Video Cleaning Failed!--->");
   }
}

void qrun()
{

   vtool();
   vinput();
   vdetect();
   ivchecker();
   vcleaning();
   vchecker();
   pause();
```

```
}

void checkrun()
{
   input();
   search();
         see();
}

void menu()
{
   int choice = 0;
   printf("\n\t|----- Menu -----|\n\n");
   printf("\n 1)Clean images\n");
   printf(" 2)Clean video\n");
         printf(" 3)Check metadata")
   while(choice != 1 && choice != 2 && choice != 3)
   {
      printf("\n Enter Your choice( 1 or 2):");
      scanf("%d", &choice);
   }
   if (choice == 1)
   {
      run();
   }
   else if(choice == 2)
   {
      qrun();
      pause();
   }
         else if(choice == 3)
   {
      checkrun();
      pause();
   }
   else
   {
      pause();
```

```
        exit(EXIT_FAILURE);
    }
}

int main()
{

    SetConsoleTitle("Metadata Remover");
    menu();
    return 0;
}
```

ДОДАТОК Б
КОД ЗАСТОСУНКУ ДЛЯ ВІДПРАВКИ ОБРОБЛЕНИХ ФАЙЛІВ

```
import smtplib
import getpass
import glob
import hashlib
from email.mime.multipart import MIMEMultipart
from email.mime.base import MIMEBase
from email.mime.text import MIMEText
from email import encoders

def authenticate(smtp, username, password):
    smtp.ehlo()
    smtp.starttls()
    smtp.ehlo()
    smtp.login(username, password)

def add_attachment(msg, filename):
    attachment = open(filename, "rb")
    part = MIMEBase('application', 'octet-stream')
    part.set_payload(attachment.read())
    encoders.encode_base64(part)
    part.add_header('Content-Disposition', "attachment; filename= %s" % filename)
    msg.attach(part)

def send_email(smtp, sender, recipient, subject, body, filenames=None):
    msg = MIMEMultipart()
    msg['From'] = sender
    msg['To'] = recipient
    msg['Subject'] = subject
```

```
    #msg.attach(MIMEText(body))

    file_hash_dict = {} # store filename and hash

    if filenames:
        for filename in filenames:
            with open(filename, 'rb') as file_to_hash:
                file_hash = hashlib.sha256(file_to_hash.read()).hexdigest()
            file_hash_dict[filename] = file_hash
            add_attachment(msg, filename)

    # append filename and hash to body
    body += "\n\n" + "\n".join(f"{name}: {hash}" for name, hash in file_hash_dict.items())
    msg.attach(MIMEText(body))

    smtp.sendmail(sender, recipient, msg.as_string())

def main():
    mailserver = 'smtp.gmail.com'
    port = 587
    sender = input("Enter your email: ")
    password = getpass.getpass("Enter your password: ")
    recipient = input("Enter recipient's email: ")
    subject = 'SMTP test'
    body = input("What to send? : ")
    needfile = input ("Want to send a file? (y/n): ")

    if needfile.lower() == "y":
        file_extension = input("Enter file extension: ")
        filenames = glob.glob('*.' + file_extension)
    else:
        filenames = None
```

```
    with smtplib.SMTP(mailserver, port) as smtp:
        authenticate(smtp, sender, password)
        send_email(smtp, sender, recipient, subject, body, filenames)

    print('Email sent!')

if __name__ == "__main__":
    main()
```

ДОДАТОК В
СПИСОК ОПУБЛІКОВАНИХ ПРАЦЬ ЗА ТЕМОЮ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Тези наукових конференцій

Костюченко В. The Crucial Role of Securing Digital Image Metadata: Protecting Privacy, Authenticity, and Integrity / Іван Пархоменко, Віталій Костюченко, Павло Горбатюк / VI Міжнародна науково-практична конференція “Проблеми кібербезпеки інформаційно-телекомунікаційних систем” (PCSITS), 2023, Київ, Україна, стр. 59-60.