Хмельницький національний університет
Факультет програмування
та комп'ютерних і телекомунікаційних систем
Кафедра кібербезпеки та комп'ютерних систем і мереж

# КВАЛІФІКАЦІЙНА РОБОТА

*бакалавр*
Освітній рівень

Система захисту від витоку даних Відділу протидії кіберзлочинам Департаменту кіберполіції Національної поліції України
Назва теми

*КвРКБ.170154.17.02.15 ПЗ*
Шифр

Галузь знань *12 «Інформаційні технології»*
Шифр, назва

Спеціальність *125 «Кібербезпека»*
Шифр, назва

Освітня програма *«Кібербезпека»*
Назва

Виконав: студент IV курсу, група КБ–17–1 _____ О.О. Стопчак
Підпис Ініціали, прізвище

Керівник _____ І.В. Муляр
Підпис, дата Ініціали, прізвище

Нормоконтролер _____ І.В. Муляр
Підпис, дата Ініціали, прізвище

До захисту допускаю:
Зав. кафедри кібербезпеки та
комп'ютерних систем і мереж _____ Ю.П. Кльоц
Підпис Ініціали, прізвище

«7» червня 2021 р.

Хмельницький 2021

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет ПРОГРАМУВАННЯ ТА КОМП'ЮТЕРНИХ І ТЕЛЕКОМУНІКАЦІЙНИХ СИСТЕМ

Кафедра КІБЕРБЕЗПЕКИ ТА КОМП'ЮТЕРНИХ СИСТЕМ І МЕРЕЖ

Освітній рівень БАКАЛАВР

Галузь знань 12 ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

Спеціальність 125 КІБЕРБЕЗПЕКА

Освітня програма ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА ПІДГОТОВКИ БАКАЛАВРІВ

ЗАТВЕРДЖУЮ

Завідувач кафедри КБКС

Клюц Ю.Г.

3 • 01 2021 р.

**ЗАВДАННЯ**
**НА КВАЛІФІКАЦІЙНУ РОБОТУ**

Стопчаку Олександру Олександровичу

Прізвище, ім'я, по батькові студента

1 Тема роботи Система захисту від витоку даних Відділу протидії кіберзлочинам Департаменту кіберполіції Національної поліції України

Керівник роботи Мулярчук Ігор Володимирович к.т.н. доцент

Прізвище, ім'я, по батькові, науковий ступінь, вчене звання

Затверджено наказом ректора університету від 05 02 2021р. № 11 додаток 9

2 Строк подання студентом роботи на кафедру: _____

3 Вихідні дані до роботи системи запобігання та виявлення витоків даних, класифікація чутливих даних, методи аналізу та виявлення чутливих даних, криптографічні засоби захисту інформації

4 Зміст пояснювальної записки (перелік питань, які потрібно розробити) Аналіз об'єкта захисту, обгрунтування вибору засобів для побудови системи безпеки, проектування системи безпеки, реалізація роботи

5 Перелік графічного матеріалу (із зазначенням обов'язкових креслень) «Алгоритм авторизації», «Алгоритм роботи модуля лексичного аналізу», «Алгоритм роботи сервера», «Архітектура системи захисту від витоків даних», «Архітектура системи захисту від витоків даних модифікована

6 Консультанти розділів курсового проекту (роботи)

| Розділ | Прізвище, ініціали та посада консультанта | Підпис, дата | |
|---|---|---|---|
| | | завдання видав | завд прий |
| Нормоконтроль | Муляр І.В., доцент кафедри КбКСМ | | |
| Антиплагіат | Муляр І.В., доцент кафедри КбКСМ | | |

7 Дата видачі завдання 05 02 2021 р.

## КАЛЕНДАРНИЙ ПЛАН

| №з/п | Назва етапів (розділів) дипломного проекту (роботи) | Термін виконання етапів проекту (роботи) |
|---|---|---|
| 1 | Вибір та затвердження теми кваліфікаційної роботи. | Січень |
| 2 | Аналіз об'єкта захисту. | Січень–лютий |
| 3 | Проектування та розробка загальної архітектури і структури системи захисту. | Лютий–березень |
| 4 | Програмна реалізація запропонованого рішення та тестування системи; аналіз результатів і оцінювання прийнятих рішень. | Квітень |
| 5 | Написання тексту пояснювальної записки та розробка графічних матеріалів. | Травень |
| 6 | Остаточне коригування кваліфікаційної роботи з урахуванням зауважень керівника. | |
| 7 | Оформлення кваліфікаційної роботи як документа відповідно до вимог. | |
| 8 | Отримання супровідних документів. Нормоконтроль. | Червень |
| 9 | Підготовка до захисту та захист кваліфікаційної роботи. | |

Студент _____ О.О. Стопчак
Підпис Ініціали, прізвище

Керівник проекту (роботи) _____ І.В. Муляр
Підпис Ініціали, прізвище

# АНОТАЦІЯ

Тема кваліфікаційної роботи: «Система захисту від витоку даних Відділу протидії кіберзлочинам Департаменту кіберполіції Національної поліції України».

Автор роботи: Стопчак Олександр Олександрович.

Керівник роботи: Муляр Ігор Володимирович.

Обсяг – 58 с., 19 рис., 2 додатка, 14 джерел.

Графічна частина: 9 презентаційних слайдів, 5 плакати.

СИСТЕМА ЗАХИСТУ ВІД ВИТОКУ ДАНИХ, ВИЯВЛЕННЯ ВИТОКІВ ДАНИХ, ЧУТЛИВІ ДАНІ.

Метою роботи є вивчення та аналіз типових проблем інформаційної безпеки пов'язаних із витоком даних, побудова системи захисту від витоку даних, що може здійснювати відслідковування чутливих даних в середині інформаційного простору компаній і виявляти неправомірні дії користувачів системи над нею.

У роботі було проаналізовано та досліджено типові проблеми інформаційної безпеки в кібернетичному просторі, особливу увагу було зосереджено на проблемі витоків даних із конфіденційною інформацією причиною якого є внутрішній порушник інформаційної безпеки.

В рамках кваліфікаційної роботи була розроблена система захисту від витоків даних, спроектована із врахуванням побажань співробітників Відділення протидії кіберзлочинам.

Підпис студента  Дата 05.06.21

| Формат | Зона | Позиц | Позначення | Найменування | Кільк. | Прим. |
|---|---|---|---|---|---|---|
| А4 | | 1 | | Завдання на дипломний проект | 1 | |
| А4 | | 2 | | Анотація | 1 | |
| А4 | | 3 | КвРКБ.170154.17.01.15 ПЗ | Система захисту від витоку даних Відділу протидії кіберзлочинам Пояснювальна записка | 1 | |
| А2 | | 4 | КвРКБ.170154.17.01.15 Е8 | Архітектура системи системи захисту від витоків даних Схема структурна | 1 | |
| А2 | | 5 | КвРКБ.170154.17.01.15 Е8 | Архітектура системи системи захисту від витоків даних модифікована Схема структурна | 1 | |
| А2 | | 6 | КвРКБ.170154.17.01.15 Е8 | Алгоритм роботи сервера Алгоритм роботи | 1 | |

| Зм. | Арк. | № Докум. | Підп. | Дата | КвРКБ.170154.17.01.15 ВП | | |
|---|---|---|---|---|---|---|---|
| Розробив | | Стопчак О.О. | | 01.06 | Система захисту від витоку даних Відділу протидії кіберзлочинам Відомість проекту | Літера | Аркуш 1 / Аркушів 2 |
| Перев. | | Муляр І.В. | | | | | |
| Н. контр. | | Муляр І.В. | | | | ХНУ, КБ-17-1 | |
| Затв. | | Кльоц Ю.П. | | | | | |

| Формат | зона | Позиц | Позначення | Найменування | Кільк. | Прим. |
|---|---|---|---|---|---|---|
| А2 | | 7 | КвРКБ.170154.17.01.15 Е8 | Алгоритм роботи | 1 | |
| | | | | модуля лексичного | | |
| | | | | аналізу | | |
| | | | | Алгоритм роботи | | |
| | | | | | | |
| А2 | | 8 | КвРКБ.170154.17.01.15 Е8 | Алгоритм авторизації | 1 | |
| | | | | Алгоритм роботи | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

| | | | | | | Арк. |
|---|---|---|---|---|---|---|
| | | | | | *КвРКБ.170154.17.01.15 ВП* | 2 |
| Зм. | Арк. | № Докум. | Підп. | Дата | | |

## ЗМІСТ



| Зм. | Аркуш | № докум. | Підпис | Дата | КвРКБ.170154.17.01.15 ПЗ | | |
|---|---|---|---|---|---|---|---|
| Розробив | | Стончак О.О. | | | Система захисту від витоку даних Відділу протидії кіберзлочинам Пояснювальна записка | Літ. | Аркуш 2 / Аркушів 58 |
| Перевірив | | Мудяр І.В. | | | | Н | |
| Н.контр. | | Мудяр І.В. | | | | ХНУ КБ-17-1 | |
| Затвер. | | Кльонін Ю.П. | | | | | |

# ВСТУП

Актуальність кваліфікаційної роботи. Розвиток науково-технічного прогресу в 90-х роках 20 століття в сфері електронно-обчислювальних механізмів дав надзвичайно сильний поштовх для розвитку нової для людини галузі – інформаційних технологій. І завдячуючи даній галузі сьогодні ми маємо новий вид суспільства – інформаційне, де інформація, яка немає власного фізичного втілення в матеріальному просторі, має надзвичайно високу ціну, хоч і в залежності від її змісту. Поява нової галузі не могла залишитися для людини непомітно, адже як принесла для нас чималі можливості з полегшенням повсякденного життя, так і принесла новий вид загроз, як і будь-яке інше досягнення науки.

Сьогодні ми маєво величезну кількість загроз пов'язаних з інформаційним простором людей, від звичайної крадіжки коштів з банківських карт до викрадення даних про розробки компанії чи держави. Такий широкий спектр загроз пов'язаний передусім із властивостями інформації, так як вона не має фізичного втілення, проте може зберігатися на матеріальних носіях, таких як магнітних дисках, оптичних носіях інформації, на папері чи у пам'яті людини. Така поліморфність інформації накладає певні особливості по її захисту в період інформаційного суспільства.

Інформаційна безпека вже досить давно стоїть на порядку денному багатьох структур. В мережевій економіці набула першочергового значення.

Інформаційна безпека має три основні складові: доступність, цілісність та конфіденційність, на забезпечення даних властивостей мусить фокусуватися будь-яка система захисту інформації. Конфіденційність визначається як забезпечення доступності інформації лише для власників інформації, та уповноваженими особами. Цілісність визначається як захист точності і повноти інформації та методів її обробки. Доступність – це можливість доступу до інформації в разі необхідності.

Конфіденційність інформації відіграє надзвичайно важливу роль. До прикладу, керівникам компанії не потрібно, щоб їх конкуренти дізналися про їх розробки чи напрямки подальшого розвитку.

Одним із прикладів порушенні конфіденційності інформації є витік інформації – це ситуація коли інформацією втрачається конфіденційність, тобто інформація стає доступною для людей які не володіють нею, і не є вповноваженими її користувачами. Доволі часто причиною витоку інформації стаю люди які авторизовані в середині мережі організації і є її користувачами. До прикладу це може бути співробітник, який був заагітований конкурентами, або скривджений внаслідок будь-якої ситуації. Доволі часто дії таких внутрішніх порушників можуть спричинити навіть більш шкідливий вплив і призвести до серйозніших втрат, ніж дії порушників ззовні.

Для вирішення проблеми витоку даних із конфіденційною інформацією, фахівцями з інформаційної безпеки було створено новий вид засобів захисту – системи захисту від витоку даних, Data Leakage Prevention System (DLP). Такі рішення фокусують на загрозах внутрішнього порушника інформаційної безпеки, тобто такого що має доступ до інформаційної системи компанії.

Подібні системи забезпечують захист інформації через відслідковування чутливою інформації серед інформаційних потоків компанії, де встановлена система захисту від витоку даних.

Мета і завдання кваліфікаційної роботи. Метою є вивчення та аналіз типових проблем інформаційної безпеки пов’язаних із витоком даних, побудова системи захисту від витоку даних, що може здійснювати відслідковування чутливих даних в середині інформаційного простору компаній і виявляти неправомірні дії користувачів системи над нею. Дана мета досягається вирішенням наступних задач:

1) Аналіз типових проблем інформаційної безпеки пов’язаних із витоком даних.

2) Розробка моделі роботи системи захисту від витоку даних.

3) Розробка алгоритму оцінки чутливої інформації.

4) Розробка програмного забезпечення згідно розроблених моделей та алгоритмів роботи.

# 1 АНАЛІЗ ОБ'ЄКТА ЗАХИСТУ

## 1.1 Історія розвитку інформаційного простору і загрози інформаційної безпеки

Поняття «інформаційна безпека» починає свою історію із моменту появи перших засобів комунікації, а також усвідомленням людиною факту завдання збитку через дії на шляхи комунікації, які здійснюються між людьми в певних товариствах спільних інтересів [4]. Традиційно виділяють наступні етапи розвитку засобів інформаційної безпеки:

— I етап – характеризується використанням примітивних засобів комунікації. Обробка та передавання даних здійснювалась ручним методом. На цьому етапі головними завданнями інформаційної безпеки було приховування та захист даних про події, факти, місцезнаходження, плани, тощо, які є життєво важливими особисто для людини або товариства до якого вона належить [1].

— II етап – обумовлений створенням перших засобів електронного зв'язку (телеграф, радіо), виникла потреба у процесах кодування/декодування інформації, а також методах ефективної передачі інформації та захисту від перехоплення (унеможливлення отримання інформації особами, для яких ця інформація не призначена). Для захисту інформації використовувались перші електромеханічні шифрувальні машини (сімейство «Енігма») .

— III етап – характеризується створенням перших електронно-обчислювальних машин (комп'ютерів). Інформаційна безпека досягалась в основному організаційними методами: обмеження фізичного доступу до засобів отримання, обробки та виводу інформації комп'ютером, так як ЕОМ були досить громісткими і займали чимале приміщення [9].

— IV етап – пов'язаний із створенням та впровадженням локальних інформаційно-телекомунікаційних мереж. Інформаційна безпека досягалась

організаційними методами та способами фізичного захисту ЕОМ, які об’єднані в локальну мережу [9].

— V етап – обумовлений широким використанням надмобільних пристроїв комунікації із широким спектром завдань і масовим розповсюдженням персональних комп’ютерів [7] в наслідок науково-технічного прогресу людства. Загрози інформаційної безпеки стали значно серйознішими та більш розповсюдженими, а також утворення плеяди людей зацікавлених у використанні цих загроз із злочинною метою – хакерів. Інформаційні ресурси стають одними із найважливіших ресурсів держави, а забезпечення інформаційної безпеки стає обов’язковою складовою національної безпеки розвиненої держави. Для повноти функціонування інформаційного простору а також забезпечення інформаційної безпеки необхідно було створити відповідні норми, правила та закони функціонування та поводження в ньому, що могли б врегульовувати усі аспекти кіберпростору. Так як специфіка функціонування інформаційного простору накладає певні обмеження, і альтернатива закону не може в повній мірі охоплювати можливі інциденти в інформаційному просторі, була сформована нова галузь міжнародної правової системи – кіберправо [6], або інформаційне право. Основним завданням даної гілки є створення нормативно-правової бази врегулювання діяльності людини в середині сучасного інформаційного простору.

В світлі розвитку інформаційних технологій, поняття інформаційної безпеки поступово розширювало своє значення. Сьогодні багато експертів замінюють поняття інформаційної безпеки іншим – кібербезпека [4]. Пов’язане це із тим, що інформаційні технології відіграють значну роль у всіх сферах життя людського суспільства, і порушення певних процесів може призвести до неминучих витрат, як часткових (незначних) так і до повної втрати контролю над певними процесами.

З урахуванням того, що проблематика кібербезпеки носить здебільшого глобальний характер, через що, важливою є позиція міжнародних організацій та установ. Так, з визначення Міжнародного союзу електрозв'язку кібербезпека – це набір засобів,принципів, стратегій забезпечення безпеки, гарантії безпеки, підходи до керування ризиками та інші засоби, які можуть бути використані для захисту кіберпростору. Основними завданнями кібербезпеки в кібернетичному просторі є забезпечення основних складових інформаційної безпеки: доступності, цілісності та конфіденційності.

Зважаючи на таку провідну роль та місце інформаційних технологій у повсякденному житті сучасного інформаційного суспільства будь-якої розвиненої держави, прийняття практично в усіх сферах курсу на інформатизацію та діджиталізацію, очевидним є факт того, що загрози інформаційної безпеки будуть актуальними і в подальшому майбутньому, а здійснення злочинів в інформаційному просторі буде зацікавлювати все більше людей [2].

Розвиток інформаційних і кібертехнологій та глобальне впровадження їх, призвели до появи нової сфери, через яку можна здійснювати різноманітні деструктивні впливи на інші сфери життя сучасної людини, суспільства чи держави. Кіберпростір доповнив існуючі сфери протистояння: сухопутна, морська, повітряна, космічна та став новою і першою в своєму роді, штучно утвореною сферою протистояння і можливих бойових дій, в тому числі здійснення тероризму, а конкретніше нового для нас кібертероризму.

Варто чітко окреслити декілька понять що будуть зустрічатися в подальшому в дані роботі:

Інформаційний простір – це сукупність результатів діяльності людства, записаний, збережений в будь-якій формі зрозумілій для людини.

Кіберпростір – це сфера, в якій застосовуються різні засоби електронної обробки, збереження, передачі, читання, перетворення, обміну інформації. Вперше поняття «кіберпростору» можна зустріти в творі «Нейромант»

американського письменника Уільяма Гібсона, використаний був для позначення всієї сукупності інформації, що міститься у комп'ютерних мережах [10].

Тобто з наступних тверджень ми можемо окреслити, що інформаційний простір – це абстракція, що під собою розуміє всі види інформації, що містить результати діяльності людини, а кіберпростір можна окреслити як частину інформаційного простору людства.

Глобальна діджиталізація суспільства та процесів виробництва призвела до загострення проблем захисту як інформації, так і самого процесу, що використовує автоматизовані системи управління для зменшення трудовитрат збоку персоналу. Генерування чималих об'ємів даних що циркулюють по глобальним мережам потребують чималих зусиль від спеціалістів із кібербезпеки, для забезпечення тріади: конфіденційності, цілісності та доступності.

Захист інформації є дещо специфічним по відношенню до аналогічних предметів, які мають фізичне втілення в природі. Дана специфіка пов'язана із особливістю інформації, вона може легко і досить швидко копіюватися, передаватись по каналах зв'язку і змінюватись. Сьогодні відомо чимало загроз інформаційної безпеки, які можуть реалізувати як зовнішні, так і внутрішні порушники безпеки.

Ефективним вирішенням проблем захисту інформації можна отримати з допомогою криптографічних методів, які дозволяють вирішувати найсуттєвіші проблеми захищеної автоматизованої обробки та передачі даних. Сучасний рівень розвитку матеріально-технічної бази та новітні методи криптографічних перетворень дозволяють забезпечити істотний рівень захищеності оброблюваної інформації без істотних втрат в продуктивності роботи таких автоматизованих систем [11, 12].

Однак так як сьогодні поняття кібербезпеки виділяють як більш розширене поняття інформаційної безпеки, не варто ігнорувати специфіку проблем

кібербезпеки. Якщо інформаційна безпека пов’язана лише із захистом виключно даних, які містять корисну інформацію, то кібербезпека розширює поле своєї діяльності на процеси обробки інформації, як автоматизовані системи будуть поводись в разі виникнення помилок пов’язаних із зміною оброблюваної інформації, а також можливий зовнішній влив інших факторів не пов’язаних із автоматизованою системою обробки та передачі інформації [14].

Сьогодні існує величезна кількість загроз кібербезпеки, серед яких сьогодні досить популярними є:

— Атаки через скомпрометовані пристрої інтернету речей. Сьогодні набуло розповсюдження різноманітних датчиків, пристроїв які мають вільне підключення до мережі інтернет. Пристрої інтернету речей стали ціллю хакерів для використання їх у ботнетах, DDoS- та Ransomware-атаках. Використовуючи прогалини в системах безпеки можна викликати збої або повне знищення техніки. Яскравим прикладом такої загрози є тест генератора Аврора, який був проведений в 2007 році. Також подібні пристрої можуть бути перепрограмовані на віддання невірних команд на ту чи іншу ситуації, або бути використані у інших цілях [7].

— Вразливості хмарних технологій. Хмарні платформи зберігають величезні об’єми цінних та конфіденційних даних, які цікаві зловмисникам. Серед загроз є вразливості Spectre та Meltdown, небезпечні (незахищені) API, а також можлива втрата даних внаслідок незловмисних причин (стихійні лиха).

— Атаки на основі машинного навчання та штучного інтелекту. Програмне забезпечення для штучного інтелекту (ШІ) та машинного навчання може «вчитися» на наслідках минулих подій, щоб досягти поставленої мети. Сьогодні фахівці з кібербезпеки широко застосовують інструменти ШІ та машинного навчання для запобігання кібератакам, але вони також не виключають можливості використання таких рішень для здійснення більш складних атак. Серед них може бути як звичайне надсилання великої кількості

спам-повідомлень, так і відгадування паролів для підготовки криптографічних атак.

— Атаки на криптовалюти та системи блокчейн. Серед компаній, які застосовують технологію крипто валют є багато таких, які нехтують впровадженням належного контролю безпеки. Як результат, це спричиняє відчутні фінансові втрати. Серед атак даного типу є: Eclipse, Sybil, DDoS атаки [1].

— Sandbox-evading Malware. Оскільки sanbox стає все більш популярним методом виявлення шкідливого програмного забезпечення, як наслідок, кіберзлочинці створюють нові штами вірусів, що можуть розпізнати, чи перебувають вони в подібних пісочницях.

— Безфайлове шкідливе програмне забезпечення – атаки, які не відносяться до шкідливого програмного забезпечення безпосередньо. Найпоширеніші такі атаки на системи під управлінням Windows, використовують вразливості операційної системи, щоб виконати своє корисне навантаження в пам’яті. Біометрична автентифікація набуває все більшої популярності як інноваційне рішення для кібербезпеки. Хоча деякі люди розглядають біометрію як новий та ефективний спосіб підвищення безпеки підприємств, інші вважають це можливою проблемою. Існує багато типів аутентифікації, заснованих на біометричних даних: загальне сканування кінчиків пальців на більш інноваційний розпізнавання голосу, райдужної оболонки ока або обличчя. Багато людей вважають, що біометричні системи практично неможливо компрометувати - дані не можна здогадатися і є унікальними для кожного користувача. Таким чином, це видається кращим рішенням для однофакторної автентифікації та чудовим доповненням до багатофакторної системи автентифікації. Однак біометричні системи мають свої недоліки. Основна проблема полягає в тому, що біометричну інформацію все одно можна вкрасти або продублювати, як і логін користувача та пароль. Однак, на відміну від пароля, користувач не може змінити сканування

райдужки або отримати нове обличчя. Це створює нові виклики для фахівців з кібербезпеки в майбутньому.

— Ransomware. Як і в попередні роки, даний вид шкідливого програмного забезпечення залишається однією з найбільш смертоносних проблем кібербезпеки. На думку багатьох експертів, програма-вимагальник стане ще гіршою в найближчі роки. Основними цілями є компанії, які зберігають цінну інформацію, таку як особисті дані користувачів або звички перегляду вебсторінок, та хмарні сервіси, особливо ті, що виконують обчислення в хмарі і, отже, зберігають величезні обсяги даних. Єдиний спосіб зменшити можливу шкоду, заподіяну цими атаками, - це резервне копіювання всіх важливих даних. Ще одним тривожним фактом є висока можливість зловмисників, які використовують методи ШІ для поліпшення своїх атак. Машинне навчання та нейронні мережі можуть використовуватися для збору конкретних даних або поширення ретельно націлених фішинг-повідомлень.

Варто окреслити ще одну загрозу інформаційної безпеки – витік даних із конфіденційною інформацією – це інцидент інформаційної безпеки, коли інформацією втрачається одна з її складових, а саме конфіденційність, тобто внаслідок будь-якого діяння чи без діяння, як навмисного, так і ненавмисного інформація з тих чи інших причин потрапляє в руки людей, що не володіють нею, і не вповноважені володарями інформації на її користування [3]. Дане визначення включає не лише зовнішній вплив, такий як втручання в систему автоматизованої обробки інформації хакерів, внаслідок якого інформація потрапляє до рук третіх осіб, але і випадки коли інформація втрачає конфіденційність внаслідок дій авторизованих в системі осіб.

Дані в кожній компанії чи організації є одним із найважливіших активів, тому захист цих даних повинен стати пріоритетною задачею. Хоча компанії мають певні заходи і програмно-апаратні засоби забезпечення безпеки в кібернетичному просторі такі, як фаєрволи, антивірусне програмне забезпечення, проте це не означає що витік інформації не може статися. Витік

інформації стається, коли чутливі дані, такі як дані кредитних карт клієнтів, персональна інформація клієнтів та співробітників, фінансова звітність, схеми апаратних розробок, інформація про дослідження, стратегії розвитку тощо, стають відкритими для сторонніх осіб, навмисно або ненавмисно. Даний інцидент інформаційної безпеки може стати величезною загрозою для подальшого функціонування компанії. Втрата конфіденційної або чутливої інформації може спричинити втрату організацією репутації, клієнтів, працівників, конкурентної переваги, втрат можливого прибутку, чи навіть до політичної кризи. Наслідки витоку даних залежать від характеру інформації, що містилась в них, та характеру діяльності компанії, організації в межах кібернетичного простору якої стався витік.

Для усунення даного виду загроз інформаційної безпеки, у 2006 році були створенні і введені в користування системи, які зосереджувалися на ідентифікації та захисті конфіденційних даних в середині автоматизованих систем обробки інформації і стали відомими як системи запобігання витоку даних. Даний вид систем захисту інформації допомагає ідентифікувати, відстежувати, захищати і знизити рівень ризику витоку конфіденційної для організації інформації. Вони використовуються для виявлення і попередження отримання неавторизованим користувачем конфіденційних даних, а також випадкового і навмисного поширення авторизованими користувачами інформації за межі інформаційно-комунікаційної мережі організації.

Витік конфіденційної інформації є серйозною небезпекою для багатьох організацій. Витік може статися в результаті навмисних дій або через необережність та необізнаність співробітників організації. Цілями навмисної організації витоку інформації може бути:

— нанесення шкоди організації, суспільству, така мета характерна для проявів кібертероризма;

— шантажування;

— отримання переваги в конкурентній боротьбі з організацією;

— помста невдоволених співробітників організації чи інших зацікавлених осіб [5].


Рисунок 1.1 - Класифікація інформації

Відповідно до класу інформації (рисунок 1.1) різняться, і шкода завдання у випадку витоку даних, що містять її, і вимоги до системи захисту організації, що оперує цими даними.

Недавні дослідження показали, що досить розповсюдженою є практика здійснення злочинів, пов'язаних із витоком конфіденційної інформації, зсередини, тобто працівниками компанії. Прикладом подібний внутрішній атак є Едвард Сноуден – американський спеціаліст в сфері інформаційних технологій та інформаційної безпеки, колишній спецагент Центрального розвідувального агентства та Агентства національної безпеки, який викрав конфіденційну інформацію, що згодом стала надбанням засобів масової інформації. Так як він мав високий рівень прав доступу до інформаційно-комунікаційної мережі як адміністратор чи менеджер, ніщо не могло запобігти отриманню ним даних, які він не повинен був отримати, без попередження нікого з офіцерів інформаційної безпеки. Даний приклад демонструє наявність вразливості і як результат завданні збитки, внаслідок реалізації загрози витоку інформації через вразливість, для конкретної організації.

Система протидії витоку даних є необхідністю для організацій, задля збереження чутливих та конфіденційних даних від витоку за межі корпоративної мережі організації через працівників, що мають авторизований доступ до цієї інформації. Будь-яка компанія має право зберігати ці дані, тому що вони є необхідними для здійснення їхньої операційної діяльності але в той же час втрата подібних даних, може поставити крапку на будь-якій діяльності компанії. Тому і розробники засобів забезпечення інформаційної безпеки, в тому числі систем протидії витокам даних, продовжують покращувати власні продукти і створювати нові розробки забезпечення безпеки в кібернетичному просторі.

Також організаціям варто звернути увагу не лише на впровадження програмно-апаратних рішень забезпечення кібербезпеки [13], а й поглянути на корінь проблеми внутрішнього порушника інформаційної безпеки, чому співробітники можу стати причиною витоку чутливих даних. Зазвичай головна причина ховається в природі людського характеру, а саме потяг до швидкого збагачення. Доволі часто співробітники можу переманюватися конкурентними

організаціями з метою отримання даних про наукові дослідження, розробки, технології, тощо. Отримання конкурентної переваги є чималим чинником у сучасній боротьбі, як окремих організацій, так і цілих країн. Різниться лише розмах боротьби і характер інформації що є метою викрадення. Ще однією важливою причиною може стати розлючення, агресія співробітника по відношенню до компанії, в якій він працює. Серед причин подібної агресії може бути як недостатня заробітна плата, так і тиск зі сторони керівництва компанії, шантажування, політичні погляди тощо. Зазвичай подібні дії можна класифікувати як прояви кібертероризму.

Варто розрізняти терміни витік інформації та її перехоплення. Перехоплення – це незаконний спосіб оволодіння даними з використанням технічних засобів. Витік даних – це втрата інформації при передачі її по каналам зв'язку і фізичному простору через усі види причин, включаючи і перехоплення інформації і її пере направлення. Навмисно створений витік інформації по технічним каналам зв'язку передбачає встановлення на шляху її розповсюдження різноманітних пристроїв, які і здійснюють її перехоплення [9].

Такий термін частіше застосовується в професійній сфері, на практиці під даним визначенням розуміються всі типи витоків інформації, засновані і на людському, і на технічному факторі. Неправомірний акт запису даних, які містять охоронювану законом таємницю, на зовнішній носій інформації і виніс його за межі корпоративного простору організації є найбільш розповсюдженим способом викрадення [10].

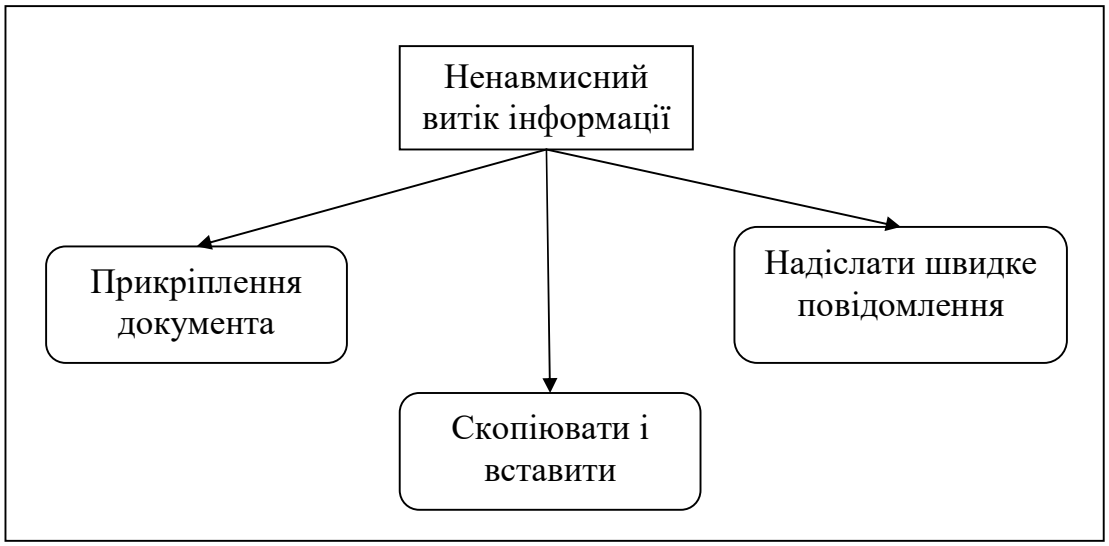



Рисунок 1.2 - Ненавмисний витік інформації

Наведений вище рисунок 1.2 показує можливі шляхи здійснення ненавмисного витоку інформації. Подібний витік зазвичай виникає через необережність самих користувачів, чи у випадках стресових ситуацій через перепрацювання, коли конфіденційна для організації інформація може надсилатися не вірному адресату чи третім особам. Це здійснюються без будь-якого злочинного наміру, проте шкоду від подібних витоків ніколи не можливо передбачити. До прикладу подібного класу витоку інформації може бути прикріплення документу, що містить конфіденційну інформацію до електронного листа і відправлення на адресу що містить помилку і імені адресата.

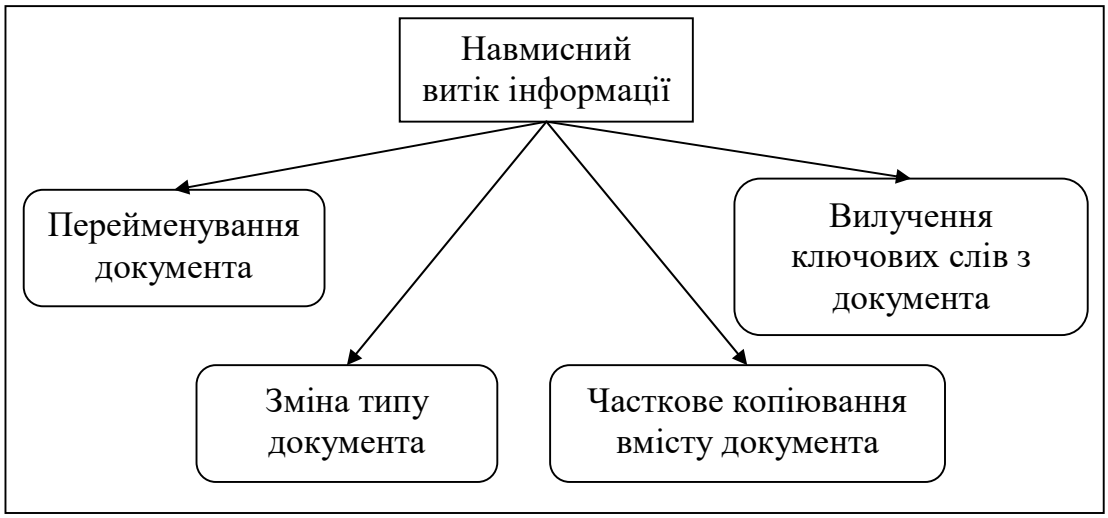



Рисунок 1.3 - Навмисний витік інформації

Навмисний витік (рисунок 1.3) інформації зазвичай відбувається коли користувач намагається надіслати документ із конфіденційною інформацією, знаючи про політику компанії, і врешті-решт все одно надсилає його в будь-який спосіб. Це робиться зазвичай коли користувач обходить правила інформаційної безпеки і нормативних актів чи пристроїв, не намагаючись отримати особисту вигоду. Наприклад подібне трапляється коли працівник перейменовує документи, змінює їх тип чи копіює вміст документу.

Зловмисний витік інформації (рисунок 1.4) вкрай сильно перекликається із навмисним витоком, і по суті своїй являється його частиною, за тим лиш виключенням що, мотивом його здійснення є отримання будь якої форми переваги, найчастіше грошової винагороди.

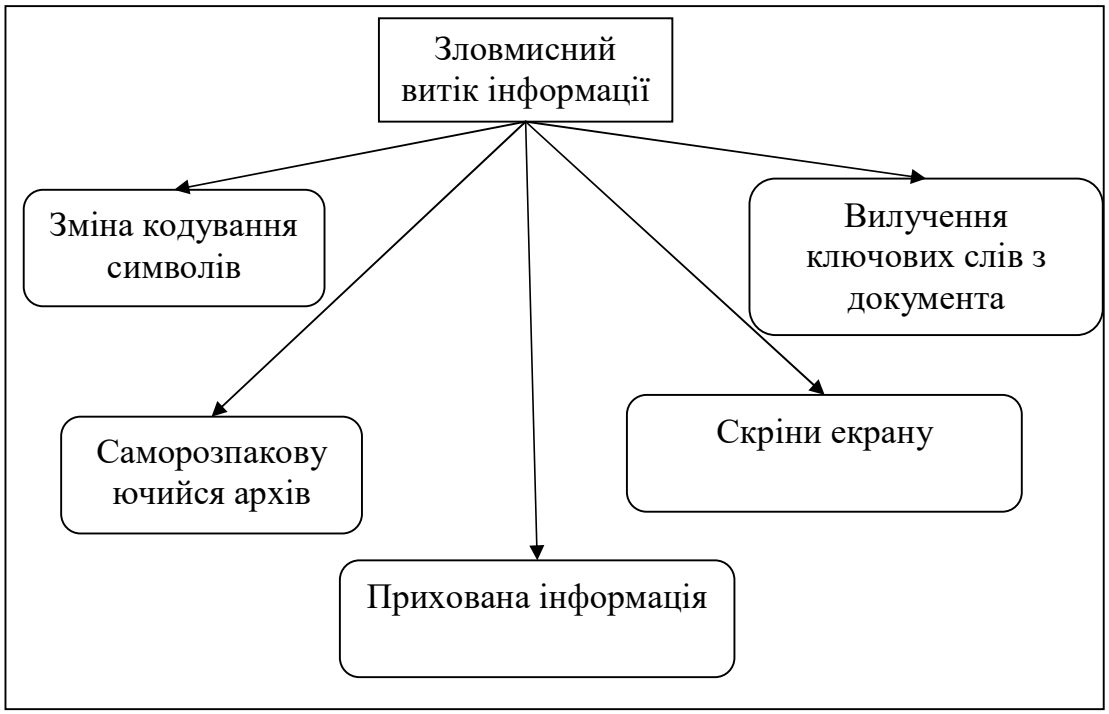



Рисунок 1.4 - Зловмисний витік інформації

## 1.2 Технології захисту даних

### 1.2.1 IDS/IPS

Система виявлення вторгнень, Intrusion Detection Systems (IDS) розробляється у вигляді пристрою, програмної реалізації або їх поєднані, який слідкує за корпоративними пристроями чи мережею на предмет дії нетипових для системи подій. Це перш за все визначає зловмисний вплив на систему, реєструє дії користувачів та стан елементів інформаційно-комунікаційної системи, намагається заблокувати і повідомити адміністраторів про підозрілу активність [10].

Система запобігання проникненню, Intrusion Prevention Systems (IPS) – це розширення системи виявлення вторгнень, оскільки обидві системи захисту здійснюють контроль над діяльністю інформаційно-комунікаційною системи на предмет підозрілого мережевого трафіку, проте система

запобіганню проникнення може зупиняти або блокувати виявлені вторгнення. Дана система також може виконувати функції попередження про потенційні небезпеки, чи вилучення шкідливих пакетів із інформаційно-комунікаційної системи компанії.

Компоненти систем виявлення/запобігання вторгнень можна розділити на два класи: мережеві системи виявлення/запобігання вторгнень та системи виявлення/запобігання вторгнень на стороні хоста.

Так перший вид систем перевіряє та контролює мережу і дані що циркулюють в ній для запобігання атакам, в той час як другий вид фокусується на аналізі трафіка на окремо взятій кінцевій точці системи, робочій станції користувача. Система виявлення/запобігання вторгнень на стороні хоста відмінно виявляє та попереджує неавторизований доступ і діяльність.

Обидва види даних систем захисту інформації можуть часто перевіряти наявність атак на систему, відстежуючи рухи зловмисників всередині мережі та попереджуючи адміністраторів про поточні атаки. Більшість подібних систем складаються з декількох додатків або апаратних пристроїв.

Системи виявлення/попередження вторгнень часто містять наступні компоненти:

— Мережеві сенсори – виявляють та надсилають дані про стан інформаційно-комунікаційної системи.

— Центральна система моніторингу – обробляє та перевіряє дані, що надходять від сенсорів системи виявлення/запобігання вторгнень.

— Аналіз звітів – компонент, який відповідає за надавання інформації про те, як реагувати на певні події кібербезпеки.

— База даних – зберігає інформацію, необхідну для роботи системи захисту, а також інформацію про зловмисника, що здійснює атаку, таку як IP-адреса тощо.

— Вікно відповідей – виводить інформації з попередніх компонентів системи і формує реакції на події.

### 1.2.2 Брандмауери

Брандмауери – це пристрої або програмне забезпечення, яке дозволяють або забороняють передачі даних та з'єднання через мережу на основі певного набору правил і використовуються для захисту мереж від несанкціонованого доступу ззовні, дозволяючи при цьому лише підключення, що дозволені адміністратором в наборі правил. Брандмауери допомагають захищати мережу від зовнішнього впливу. Його основне завдання полягає в контролі вхідного та вихідного трафіку мережі шляхом аналізу пакетів даних, та визначення того чи повинен пакет бути дозволеним чи забороненим [12].

### 1.2.4 SOC

Для забезпечення інформаційної безпеки на підприємстві існує рішення із створенням нової структурної одиниці всередині організації, для здійснення процесу забезпечення і підтримки захищеності організації від загроз в інформаційному просторі. Одним із варіантів є введення операційного центру безпеки (Security Operations Center або SOC) – це організаційно-штатна структура організації, що відповідає за моніторинг та аналіз стану інформаційної безпеки організації, в тому числі виявлення, аналіз та реагування на інциденти кібербезпеки, з допомогою поєднання технологічних та організаційних рішень. В сферу діяльності даного структурного об'єкта входить також безперервний моніторинг, контроль, оцінка та захист усіх елементів автоматизованої системи обробки інформації організації, а саме веб-сайтів, баз даних, додатків, серверів,

мережевого обладнання, пристроїв «інтернету речей», систем охорони, комп’ютерів та іншого обладнання [12].

Операційний центр безпеки відстежує та аналізують діяльність компонентів інформаційної системи організації, шукаючи в них аномальну активність, що може свідчити про інцидент інформаційної безпеки чи збій роботи.

Замість розробки стратегії безпеки, конструюванні архітектури інформаційної безпеки чи реалізації заходів пов’язаних з кібербезпекою, команда операційного центру безпеки несе відповідальність за постійний, оперативний компонент інформаційної безпеки підприємства. Співробітники центру операційної безпеки в першу чергу складають аналітики інформаційної безпеки, які працюють разом за для виявлення, аналізу, реагування, звітуванням та запобіганням інцидентам інформаційної безпеки. В окремих випадках SOC може включати вдосконалений криміналістичний аналіз, крипто аналіз та реверсивне проектування шкідливого програмного забезпечення для покращеного аналізу інцидентів кібербезпеки.

## 1.3 Висновки

В ході проведення збору та систематизації інформації по предметній області кваліфікаційної роботи була виявлена взаємозалежність між рівним поширення і розвитку інформаційних технологій та кількістю загроз в кібернетичному просторі. Аналіз підходів до забезпечення інформаційної безпеки протягом розвитку IT дав підґрунтя для синтезування нових підходів у забезпеченні інформаційної безпеки з урахуванням проблем попередніх рішень безпеки.

Аналіз типової архітектури систем запобігання витоків даних дозволив синтезувати власний підхід до виявлення і аналізу інцидентів кібербезпеки, пов’язаних із витоком даних.

Так як розвиток інформаційних технологій продовжує розвиватися і розширювати власні межі охоплення господарської та повсякденної діяльності людини, інтегруючи різноманітні програмно-апаратні рішення у найрізноманітніші сфери. Дана перспектива розвитку продовжить підживлювати інтерес у людей із злочинними намірами до сфери інформаційних технологій, що свою чергу продовжить породжувати все нові проблеми в кібернетичному просторі людства.

# 2 ОБҐРУТНУВАННЯ ВИБОРУ ЗАСОБІВ ДЛЯ ПОБУДОВИ СИСТЕМИ БЕЗПЕКИ

## 2.1 Опис мови програмування C#

Для розробки «Системи захисту від витоків даних» мною була обрана мова програмування C#. Це сучасна мова програмування із жорстокою типізацією. Написання програм даною мовою здійснюється з використанням парадигми об’єктно-орієнтованого програмування. C# забезпечую мовні конструкція для підтримки та реалізації концепцій об’єктно-орієнтованого програмування, а також компонентно-орієнтованого. C# починає своє коріння з сімейства мов C, на що натякає назва.

Від попередників мову C# відрізняє величезна кількість нововведень, які значно допомагають розробникам програмного забезпечення і полегшують процес розробки та відлагодження програм. Серед них є:

— Збирач сміття, який автоматично відновлює пам’ять, яка зайнята недосяжними об’єктами, тобто такими, які вже не використовуються програмою.

— Nullable типи, що служать для захисту від змінних, що не містять посилання на об’єкти.

— Обробка винятків забезпечує структурований підхід до виявлення та оброблення помилок в ході виконання додатків.

— Лямбда-вирази підтримують функції функціонального програмування та створені анонімних функцій.

— LINQ – мова інтегрованих мовних запитів, допомагає із маніпулюванням структурованих даних будь-якої форми і будь-якого джерела.

— Підтримка асинхронного програмування забезпечує легке маніпулювання потоками в середині програми.

Програми написані мовою C# працюють на платформі .Net, у віртуальному середовищі виконання, яка називається common language runtime (CLR) та набором бібліотек класів.

CLR – це впроваджена компанією Майкрософт загальна мовна інфраструктура, в якості міжнародного стандарту. Загальна мовна інфраструктура є базовою основою для створення середовища розробки і виконання.

Ще одною перевагою мови C# є можливість розробки на основі платформи .Net Core. Дана платформа є розробленою на основі класичного .Net Framework, про на відміно від нього є модульною платформою для розробки із відкритим кодом. Дана платформа є сумісною з білістю операційних систем і є кроссплатформенною.

## 2.2 Опис системи управління базами даних

Для зберігання даних необхідних для роботи розроблюваної системи було обрано систему управління базами даних MS SQL Server. Дана СУБД є однією з найбільш популярних для персонального та корпоративного користування. Відноситься до реляційних розподілених клієнт-серверних систем управління базами даних.

Головними перевагами є:

— Надійсть та безпека. MS SQL Server надає інструменти для шифрування даних.

— Продуктивність.

— Легкість у користування та адмініструванні.

Основна мова, яка використовується для роботи це Transact-SQL – розробка компаній Майкрософт та Sybase.

Для роботи із MS SQL Server з вебсерверу використовується об’єктно-реляційна технологія відображення (ORM) – Entity Framework Core. Дана технологія дозволяє абстрагуватися від структури сховища даних і працювати із даними таблиці бази даних, ніби вони були об’єктами всередині додатку. Тобто якщо на фізичному рівні ми керуємо таблицями, індексами, первинними і вторинними ключами, то на концептуальному рівні, який надається нам технологією Entity Framework Core, ми керуємось об’єктами.

Основною причиною вибору даної ORM системи є використання запитів LINQ для вибірки даних із сховища та можливість оптимізації цих запитів. З допомогою LINQ розробник може створювати також різноманітні запити на вибірку об’єктів, які є зв’язаними різноманітними видами реляційних зв’язків.

## 2.3 Опис вебсерверу

Вебсервер – це програма, яка служить для обробки запитів по протоколу HTTP, і відправляє відповіді на них. Служить для зберігання, зчитування, обробки, передавання інформації клієнтів вебсерверу. Клієнтами вебсерверу можуть бути браузери, інші вебсервери, програми і пристрої.

Для побудови вебсерверу, що міг би здійснювати керування системою і став його ядром, я обрав платформу ASP.NET Core від компанії Майкрософт. Призначення даної платформи полягає у створені веб-додатків різного профілю, як звичайних сторінок магазинів, так і для проектів із складною логікою.

ASP.NET Core може працювати поверх крос-платформної середовища .NET Core, яка може бути розгорнута на популярних операційних системах: Windows, Mac OS, Linux. І таким чином, за допомогою платформи ASP.NET Core ми можемо створювати крос-платформні додатки. І хоча Windows як середовище для розробки додатків на даній платформі і розгортання програми досі превалює, але тепер вже ми не обмежені тільки даною операційною

системою. Тобто ми можемо запускати вебдодатки не тільки на ОС Windows, але і на Linux і Mac OS. А для розгортання веб-додатки можна використовувати традиційний IIS, або крос-платформний вебсервер Kestrel.

Для розробки користувацького інтерфейсу керуванні системою був використаний архітектурний шаблон проектування модель-представлення-контролер. Даний шаблон проектування розподіляє додаток на 3 частини: модель даних, представлення та контролер. Застосовується для відокремлення даних від інтерфейсу користувача так, щоби зміни в одному елементі мінімально впливали на інші елементи.

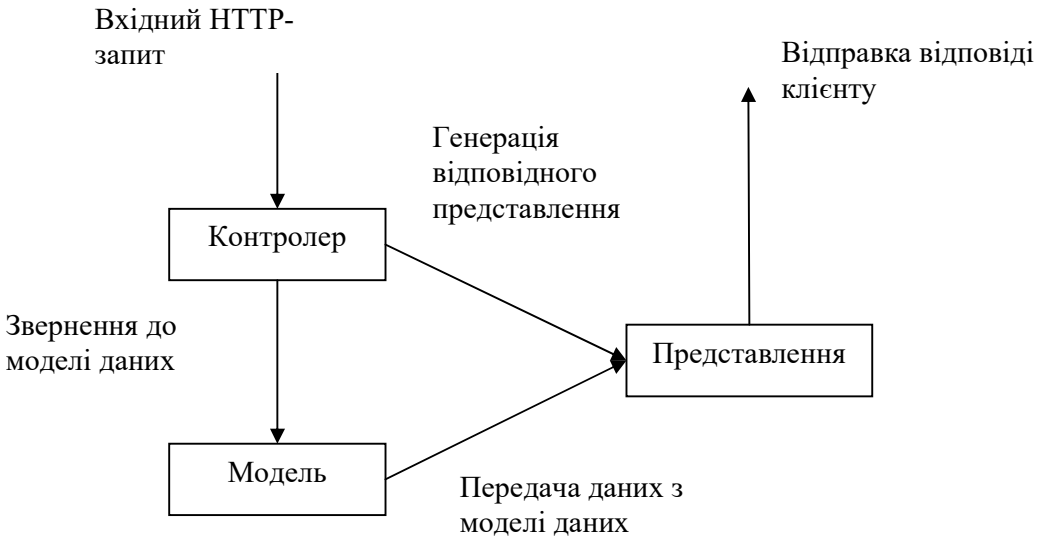



Рисунок 2.1 - Шаблон модель-представлення-контролер

Дане розділення компонентів додатку (рисунок 2.1) дозволяє реалізувати концепцію розподілення відповідальності, де кожен елемент відповідає за власну чітко окреслену сферу. Як наслідок розробнику простіше здійснювати розробку, підтримку та тестування окремих компонентів.

На даному рисунку 2.1 модель є повністю незалежним структурним елементом, а контролер і представлення є відносно незалежними. Тобто будь-які зміни в контролері чи представлені ніяк не вплинуть та модель даних, проте

в більшості випадків зміни в моделі несуть за собою зміни в інших елементах шаблону.

Для передавання повідомлень між клієнтами та вебсервером буде використовуватися технологія SignalR. Для реалізації даної технології на платформі .Net компанією Майкрософт була створена бібліотека SignalR Core. Дана технологія використовується для побудови додатків, працюючих в режимі реального часу, що є основною вимогою для побудови нашої системи. SignalR створює двонаправлений зв'язок між клієнтом та сервером, через який здійснюється передача певних даних і команд в режимі реального часу.

Основною одиницею в даній технології є хаб, до якого здійснюють підключення клієнти. В межах хабу здійснюється вся логіка обробки повідомлень клієнтів та надсилання відповідей на основі певної бізнес логіки.

## 2.4 Середовище розробки

Для розробки програмних складових системи було обране інтегроване середовище розробки Visual Studio Community – досить потужний інструмент для редагування, налагоджування та побудови додатків. Крім стандартного редактора коду та відлагоджувача, які надаються більшістю середовищ розробки, в інструментарій входять компілятори, засоби рефакторінга коду, графічні дизайнери, очищення коду, пошуку та навігації всередині проектів, довідкові функції IntelliSense та інші.

Сукупність цих факторі, а також попередня знайомство із даним інтегрованим середовищем розробки вплинуло на вибір даного інструменту в рамках даної кваліфікаційної роботи.

# 3 ПРОЕКТУВАННЯ СИСТЕМИ БЕЗПЕКИ

## 3.1 Система запобігання витоків даних

### 3.1.1 Типова архітектура

Головною метою системи запобігання  витоків даних є захист чутливих даних на різних фазах, загальна структуру можна переглянути на рисунку 3.1.

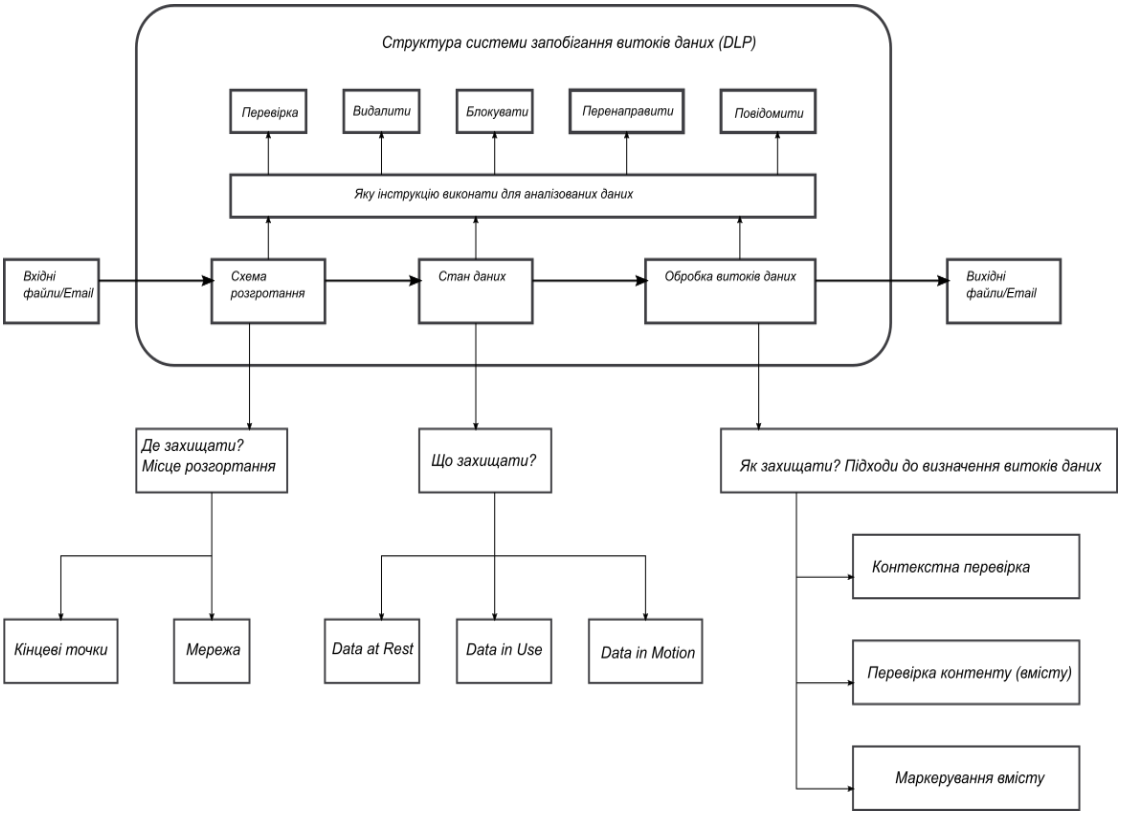



Рисунок 3.1 - Структура роботи системи запобігання витоку даних

Системи DLP розділяють три стани даних протягом їх життєвого циклу:

— Data-at-rest (DAR), дані у стані спокою – категорія, яка описується як усі дані, що зберігаються на комп'ютері. Для уникнення неправомірного доступу до даних, їх викрадення чи модифікації сторонніми особами, зазвичай застосовуються заходи інформаційної безпеки, такі як, шифрування даних та контроль доступу. Обов'язковою передумовою для здійснення даних заходів безпеки є виявлення контенту, який служить для виявлення місця зберігання даних. Найкращий спосіб досягнення даної мети – це застосування функцій виявлення та аналізу вмісту системи запобігання витоків даних. Наприклад, політика інформаційної безпеки може вимагати, щоби дані кредитних карток, дані клієнтів зберігалися лише на затверджених серверах. У випадку якщо дані були виявлені на незатвердженому сервері, вони можуть бути зашифровані або видалені, або власнику даних може бути надіслано попередження.

— Data-in-motion (DIM), дані в русі – дані, які надсилаються через мережу. Цей вид даних може передаватися як в середині інформаційно-комунікаційної системи організації, так і може надходити у глобальну (зовнішню) мережу. Системи запобіганню витоків використовують для виявлення та аналізу даних, що передаються з допомогою протоколів (рисунок 3.2) по каналам зв'язку, так як SMTP (простий протокол передачі пошти), HTTP (протокол передачі гіпертексту), FTP (протокол передачі файлів), тощо

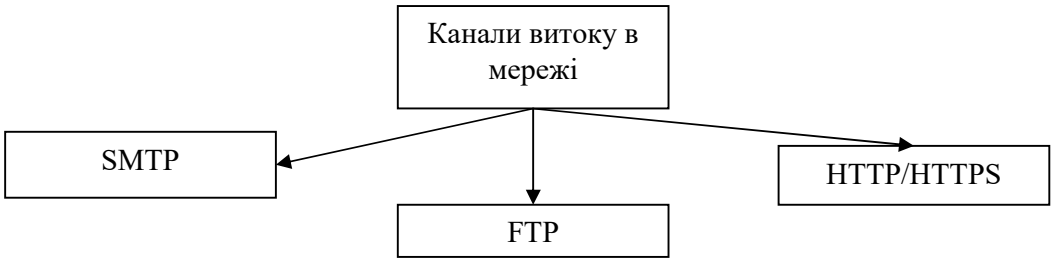



Рисунок 3.2 - Канали витоку даних в мережі

— Data-in-use (DIU), дані у користуванні – це будь-які дані, які користувач використовує у здійсненні своєї діяльності. Системи запобігання витоків даних, орієнтовані на кінцеві точки, використовуються безпосередньо для захисту даних у користуванні, та для здійснення контролю над ними під час взаємодії користувача з ними. Зазвичай система використовується для моніторингу чутливих даних під час їх використання або передавання з пристрою кінцевої точки або клієнта через різноманітні вихідні канали на периферійні пристрої (рисунок 3.3). Фактична концепція полягає в тому, що коли робиться спроба надіслати конфіденційні дані, потенційні витоки даних можуть бути швидко виявлені та усунені, шляхом блокування, перш ніж дані будуть надіслані стороннім особам. Інструменти даного підвиду систем націлені на здійснення контрою за наступними видами активностей:

a) Операції копіювання, передачі і здійснення знімків екрану, що виконуються над даними із конфіденційною інформацією.

b) Передача конфіденційної інформації з одного пристрою на інший, наприклад, USB-накопичувачі, CD/DVD диски або смартфони.

c) Друкування конфіденційних даних.

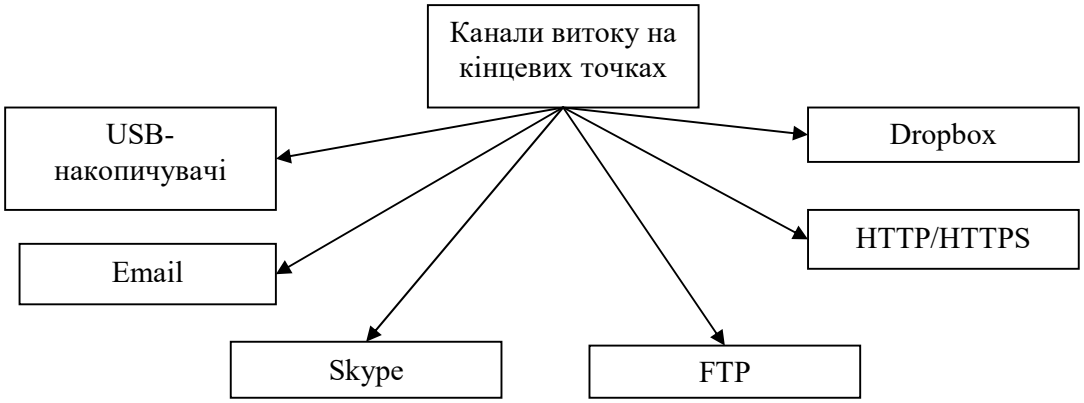



Рисунок 3.3 - Канали витоку на кінцевих точках

### 3.1.2 Підходи до аналізу витоків

Для запобігання витоку даних із конфіденційною інформації, його потрібно перш за все виявити. Для здійснення виявлення було розроблено декілька можливих підходів:

— Перевірка на основі контексту. Дана модель здійснює контроль над безліччю систем безпеки, таких як брандмауери, проксі-сервери, системи виявлення/запобігання вторгнень та фільтрацію спаму. Головна ідея полягає у відслідковуванні контекстної інформації про дані, що передаються в середині інформаційно-комунікаційної мережі, наприклад відправник, отримувач, розмір, призначення, заголовки, метадані, типи файлів тощо. Зразком даного підходу, заснованої на аналізі контексту даних, є фільтр пакетів брандмауера, який визначає, чи може пакет даних продовжувати свій шлях по мережі. Основою для фільтрації пакетів може бути IP адреса відправника/отримувача, порти та інші атрибути пакетів даних.

— Перевірка на основі вмісту. Даний підхід до аналізу витоків даних аналізує вміст, використовуючи ряд методів, таких як:

a) Аналіз на основі словника ключових слів. Наприклад подібним ключовими словами можуть бути «політика», «пароль», «зарплата», тощо. Характер ключових слів перш за все залежить від сфери діяльності організації. Також ключовими можуть бути шаблони на відповідність, до прикладу 16-значний номер кредитної картки тощо. Багато систем включають узагальнені словники, які стосуються нормативних актів та законів, наприклад PCI-DSS (Стандарт безпеки даних платіних карток), HIPAA (Закон про переносимість та підзвітність медичного страхування) тощо. Даний метод є найбільш простим і найбільш швидким у застосуванні.

b) Метод зняття цифрових зліпків. Даний підхід бере своєрідні цифрові «відбитки пальців» із конфіденційних файлів та записів баз даних і шукає відповідні відбитки для виявлення витоків даних. Даний відбиток є унікальним

хеш-значенням, що буде відповідати лише даному набору даних. Значення хешу конфіденційних даних зберігається локально на контрольованому пристрої, або в базі даних. В подальшому система порівнює значення хешів і на основі результатів здійснює оцінку.

c) Метод навчання. Основною ідею даного підходу є використання методів машинного навчання, таких як Vector Space Model (VSM), для визначення «рівня конфіденційності» аналізованих системою даних.

— Маркування вмісту. Основна мета даного підходу полягає в маркуванні вмісту файлу з конфіденційною інформацією. В подальшому вміст залишатиметься із даними позначками навіть після обробки його іншими програмами. Наприклад позначений файл залишатиметься таким навіть після шифрування його вмісту, перекодуванні його чи стиснені в архів.

## 3.2 Технічне завдання на розроблювану систему

### 3.2.1 Загальні відомості

Назва розроблюваної системи: «Система захисту від виток даних Відділу протидії кіберзлочинам Департаменту Кіберполіції Національної поліції України».

### 3.2.2 Мета і призначення розроблюваної системи

Основною метою даної розроблювана системи є введення превентивних мір по забезпечені захисту від несанкціонованих витоків інформації в інформаційно-комунікаційному середовищі відділення протидії кіберзлочинам.

Додатковою метою створення даної системи є впровадження власного продукту який міг задовольнити потреби захисту від витоків інформації, і в

той же час впродовж своєї експлуатації зміг би розширити спектр надаваних функцій.

Системи призначена для виявлення, сигналізування та запобігання несанкціонованому використанню, передачі даних, що містять конфіденційну інформацію.

Сферою застосування є забезпечення захисту від інцидентів інформаційної безпеки пов’язаних із несанкціонованим доступом до інформації та витоками даних із конфіденційною інформацією.

### 3.2.3 Загальна характеристика розроблюваної системи

В рамках проект потрібно реалізувати сервер, серед функцій якого буде:

— Можливість додати документ з конфіденційною інформацію на відслідковування.

— Конфіденційна інформація не має зберігатися на сервері.

— Сервер може бути виконаний у вигляді вебсайту.

— Може отримувати повідомлення від клієнтів із даними що циркулюють на кінцевих точках.

— На основі даних з повідомлення проводиться лінгвістичний аналіз інформації.

— На основі проведеного аналізу здійснюється оцінка чи містить дані, що передаються користувачем, конфіденційну інформацію організації.

— Надіслання повідомлення-відповіді на основі оцінки.

— Архітектура серверу двохланкова.

— Сервер повинен бути абстрагований від структури бази даних, що міститиме необхідні для його роботи дані.

— Повинна бути реалізована рольова політика для розмежування доступу серверу.

— Повинні бути реалізована класифікація інформація по рівню критичності її витоку для організації.

— Реалізувати можливість надавання ключів для операції шифрування/дешифрування користувачам.

— Надавання ключів повинно базуватися на ролі користувача і політиці розмежування доступу організації.

— Сервер має бути незалежним від платформи.

— Має містити функціонал для звітування офіцерам інформаційної безпеки.

— Повинен повідомляти офіцерів безпеки про інциденти кібербезпеки.

Також для здійснення контролю за циркулюючою інформацією на кінцевих точках (робочих станціях користувачів) потрібно реалізувати клієнтський додаток, серед функцій якого має бути:

— Самозавантаження в операційній системі.

— Повинен працювати на будь-яких платформах і операційних системах.

— Відслідковувати підключення зовнішніх пристроїв.

— Відслідковувати дані що знаходяться в буфері обміну.

— Надсилати повідомлення з інформацією, що оперується на робочому місці користувача.

— Запобігати витоку інформації на основі оцінки отриманої із сервера.

— Містити функціонал для авторизації користувача.

— Мати функцію для здійснення операцій шифрування та дешифрування файлів.

## 3.3 Опис функціонування системи

Згідно із сформованим технічним завданням, необхідно описати моделі потоків даних та схему роботи розроблюваної системи. Разом із цим потрібно описати інструменти підтримки та адміністрування даної системи.

Розробка системи захисту від витоків даних буде виконуватися згідно із клієнт-серверною архітектурою (рисунок 3.4). Дана архітектура передбачає розподілення функцій системи між провайдерами послуг та їх клієнтами. Фактично реалізацією провайдера та клієнта розроблюваної системи може бути програмне забезпечення. Класично провайдера послуг називають сервер. Основною формою взаємодії клієнта та сервера є надсилання запитів через мережеве середовище зв'язку.

Клієнт – це додаток на робочій станції користувача, який надсилає запити на сервер і на основі відповідей серверу виконує певні інструкції.

Сервер – додаток на спеціалізованій робочій станції, яка налаштована для отримання запитів від багатьох клієнтів та обробки їх згідно власної логіки роботи і надсилання відповіді на основі результатів обробки.

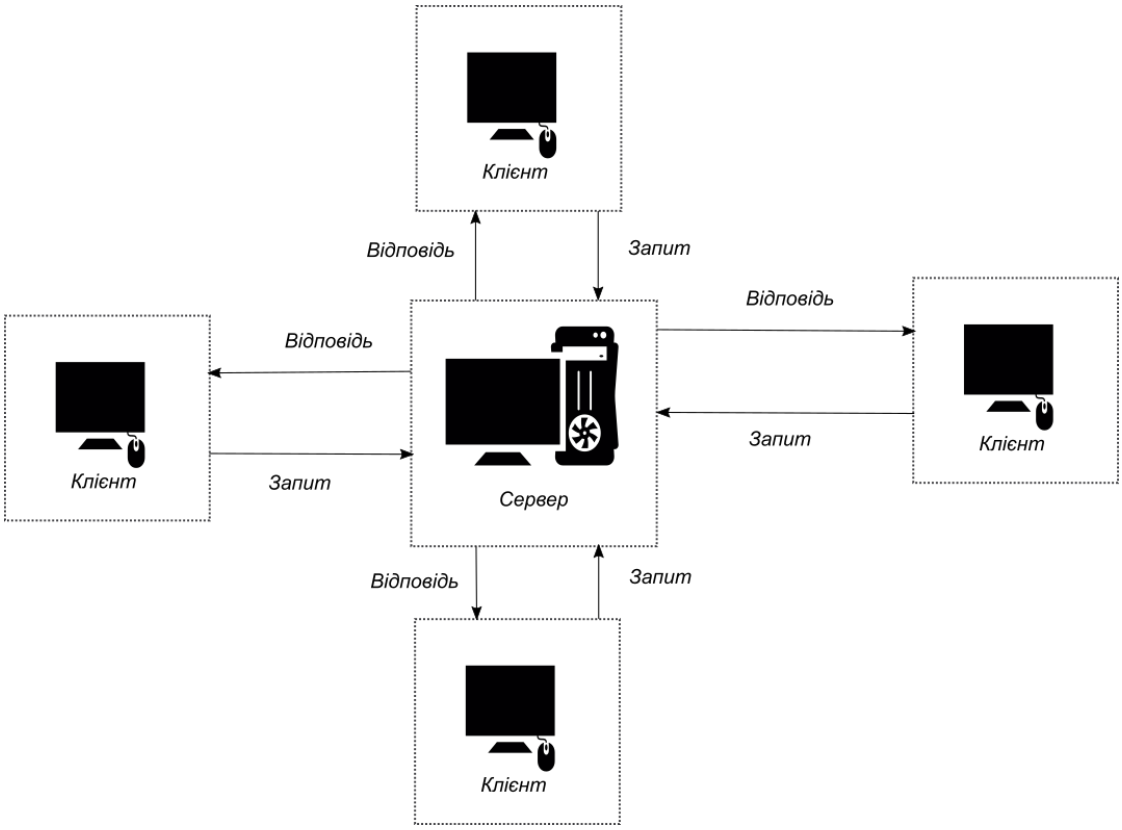



Рисунок 3.4 - Клієнт-серверна архітектура

Функціональні можливості серверу та клієнту повинні відповідати технічному завданню, наведеному в пункті 3.2.3

Для здійснення адміністрування над системою і додаванням файлів на відслідковування. Даний модуль контроль буде виконаний у вигляді вебпанелі (рисунок 3.5) і складатиметься з таких частин:

1) Меню.
2) Робоча зона.

В меню міститимуться такі пункти:

1) Керування інформацією, що відслідковується.
2) Керування користувачами системи.
3) Керування ключами криптографічного захисту інформації.

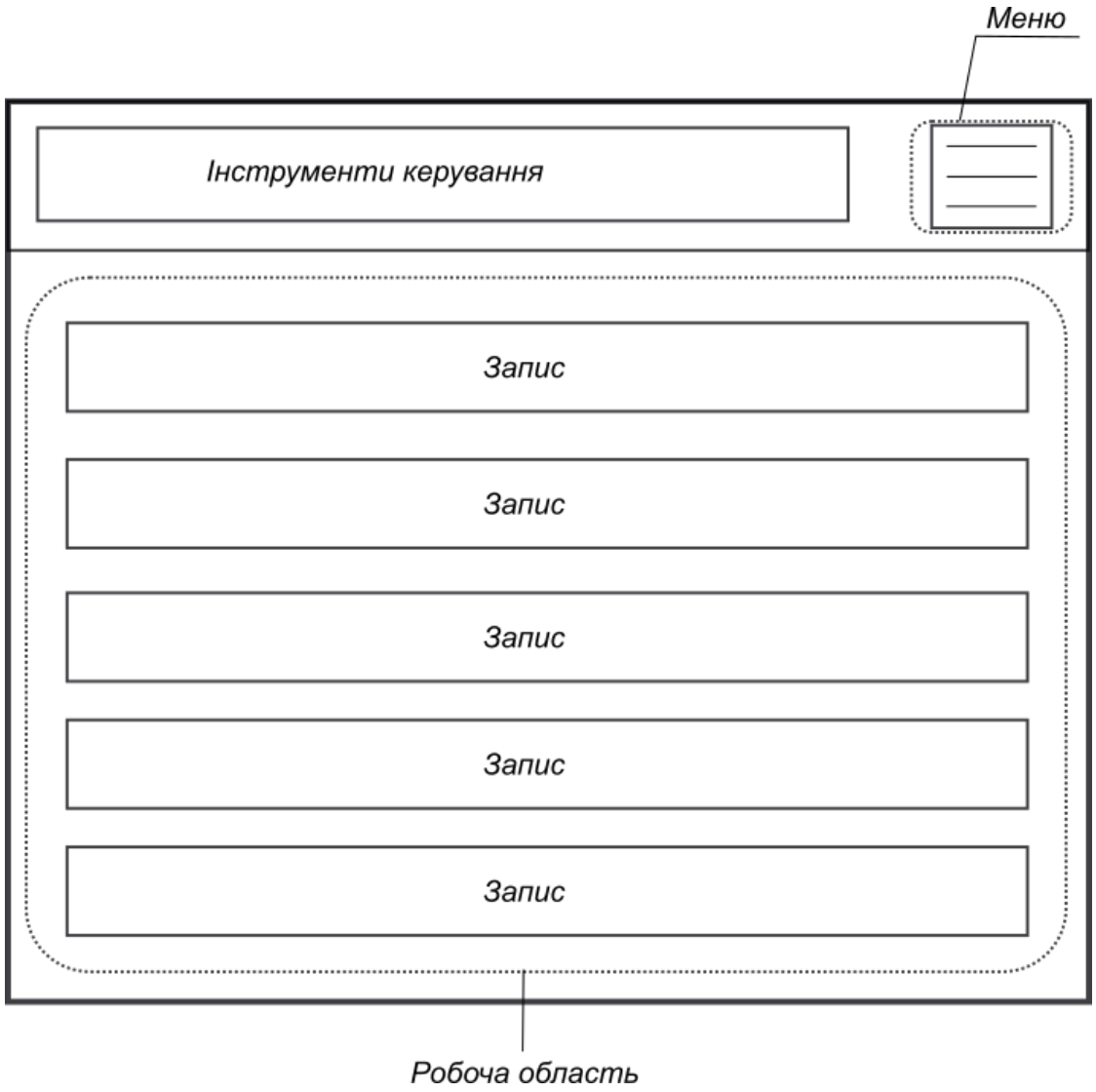



Рисунок 3.5 - Модель інтерфейсу вебпанелі керування системою

Робоча зона міститиме функціонал в залежності від вибору пункту меню.

При виборі пункту керування інформацією, що відслідковується повинна бути можливість завантаження файлу або додавання тексту, який в подальшому пройде послідовно процеси нормалізація тексту, розбивання нормалізованого тексту на лексеми і токенування лексем і запису результатів до сховища даних системи. Конфіденційна інформація в середині системи не зберігається, лише її цифрові відбитки у вигляді мішка слів (токенів). Серед інших функцій має бути можливість видалення неактуальних для захисту даних, та виставлення пріоритетів захисту на інформацію.

В пункті керування користувачами системи здійснюються операції додавання, редагування та видалення користувачів, керування ролями користувачів в системі.

При виборі керування криптографічними засобами захисту інформації в робочій області висвітлюються ключі алгоритмів шифрування, які були застосовані до файлів із конфіденційною інформацією і можуть бути надані користувачам системи згідно із політикою безпеки і правами доступу користувача.

### 3.4 Розробка структурної схеми системи

Для реалізації даної системи захисту від витоків пропонується наступна загальна архітектура система, зображена на рисунку 3.6. Ядром керування нашої систему буде вебсервер, який через API буде взаємодіяти з іншими елементами системи захисту від витоку даних, а саме із клієнтами на робочих станціях. Дана система інтегрується в інформаційно-комунікаційне середовище та здійснює відслідковування інцидентів інформаційної безпеки пов’язаних із несанкціонованою передачею конфіденційної інформації.

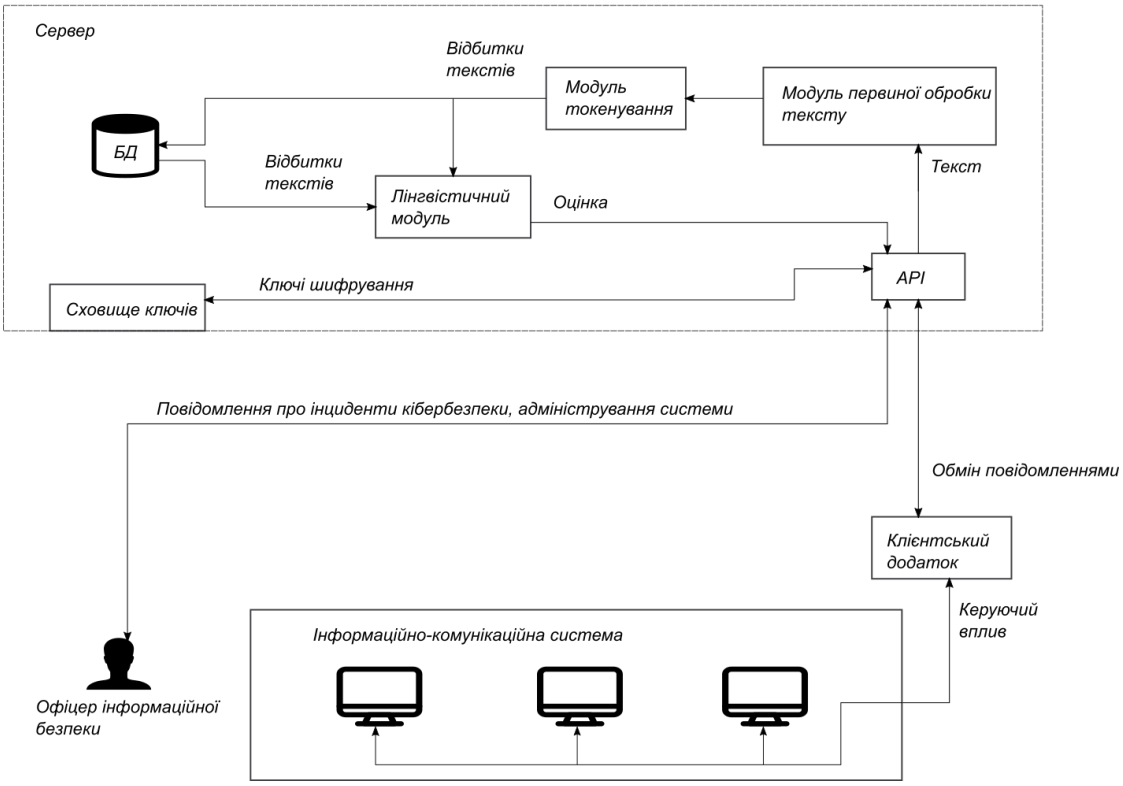



Рисунок 3.6 - Архітектура системи захисту від витоку даних

Серед внутрішніх компонентів серверу є:

1) БД – база даних, що міститиме цифрові «зліпки» конфіденційної інформації, а також дані про користувачів системи.

2) Сховище ключів – елемент, що міститиме криптографічні ключі. Виділено у окремий, так як необхідно реалізувати спеціалізований захищений інтерфейс взаємодії.

3) Лінгвістичний модуль – здійснює аналіз цифрових відбитків текстів із записаними в базі даних. На основі проведеного аналізу дає оцінку конфіденційності аналізованого тексту.

4) Модуль первинної обробки тексту – компонент, який відповідає за очищення тексту від малозначимих символів та розділових знаків, а також здійснює нормалізацію вхідних текстів.

5) Модель токенування – здійснює зняття цифрового «відбитку» текстів, для подальшого його лексичного аналізу.

6) API – інтерфейс взаємодії із внутрішніми компонентами серверу.

Для повноти проектної інформації необхідно також окреслити архітектуру додатку, який буде виконувати відслідковування потоків даних, що передаються на конкретній робочій станції і виконувати додаткові задачі із забезпеченням захисту конфіденційних даних.

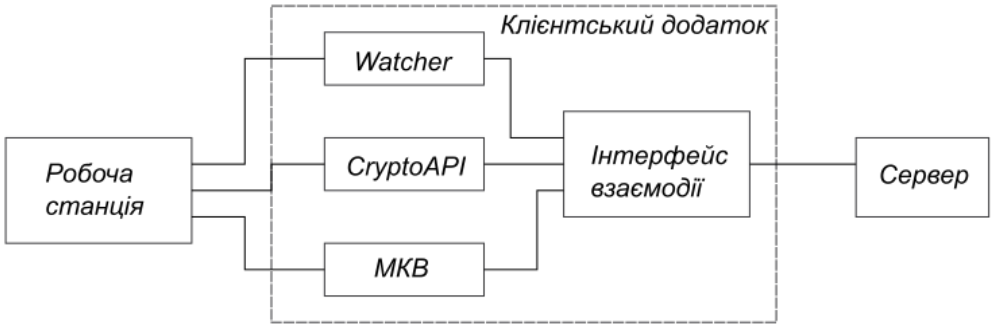



Рисунок 3.7 - Архітектура клієнтського додатка

На рисунку 3.7 представлена схема клієнтського додатку, серед модулів якого є:

1) Watcher – модуль, який буде здійснювати відслідковування файлової системи на предмет неправомірних дій над файлами із конфіденційною інформацією, а також відслідковування інших шляхів витоку даних.

2) CryptoAPI – компонент системи захисту, що буде здійснювати операції шифрування та дешифрування файлів, на основі наданих сервером ключів криптографічного захисту інформації.

3) МКВ – модуль керуючого впливу, на основі команд сервера здійснює контроль над потоками робочої станції.

Також на схемі представлені:

1) Робоча станція над якою здійснюється захист від витоку даних.

2) Сервер, який здійснює контроль за станами клієнтського додатку. Відповідно до команд серверу модуль керуючого впливу здійснює керування потоками даних робочої станції.

В даної архітектури є головний недолік – це пересилання аналізованого тексту по каналам мережевого зв'язку, що в свою чергу призводить до збільшення навантаження на інформаційно-комунікаційну мережу, а також з'являється ризик перехоплення конфіденційної інформації.

Для вирішення даної проблеми є два підходи:

1) Шифрування тексту, який передається на аналіз серверу

2) Перенесення модулів токенування та первинної обробки тексту із сервера на клієнт.

Перший підхід вирішить проблему перехоплення інформації, але водночас створить додаткове навантаження на клієнт та сервер і не вирішиться проблема навантаження мережевого середовища.

Другий підхід дозволить знизити навантаження на сервер і на мережеве середовища. І так як передаватися через мережу буде множина токенів, це частково вирішить проблему перехоплення інформації.

Для вирішення даної проблематики мною був обраний другий підхід. Звідси архітектура системи захисту від витоку даних набуває наступного вигляду (рисунок 3.8).

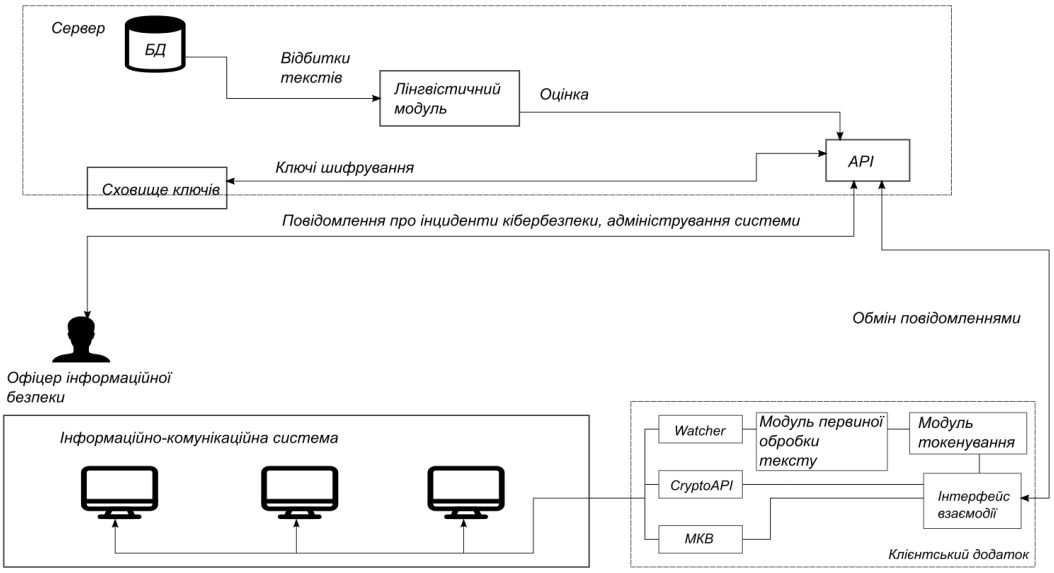



Рисунок 3.8 - Модифікована архітектура системи захисту від витоку даних

## 3.4 Висновки

В ході проектної розробки було створено структуру майбутньої системи захисту від витоків даних, сформоване технічне завдання на розроблювану схему. Для ефективної роботи системи захисту від витоків даних необхідно в структурну схему системи ввести компонент лексичного аналізу тексту. Для порівнювання і аналізу текстів будуть використовуватися цифрові «відбитки» текстів або файлів із конфіденційною інформацією, для подальшого її відслідковування в інформаційно-комунікаційній системі компанії.

Сформовано принципи роботи системи захисту від витоків даних для забезпечення безпеки конфіденційної інформації і унеможливлення ризику витоку даних через структуру системи.

На стадії проектної розробки було виявлено значний недолік пов’язаний із передачею незахищеної інформації по інформаційно-комунікаційному середовищі і навантаження на нього через можливу передачу великих об’ємів даних. Вирішення даної проблеми було досягнуто за рахунок модифікації архітектури розроблюваної системи.

# 4 РЕАЛІЗАЦІЯ РОБОТИ

## 4.1 Розробка блок-схем і опис алгоритмів функціонування системи

На основі описаного в 3 розділі технічного завдання та розробленої архітектури майбутньої системи захисту від витоків даних приступаємо до розробки та реалізації алгоритмів функціонування системи.

Розглянемо алгоритм, зображений на рисунку 4.1, роботи компонента обробки запитів клієнтів системи захисту від витоків даних до серверу. Першим етапом роботи системи є розгортання елементів необхідних для роботи системи, таких як:

— Контекст даних бібліотеки Entity Framework Core, підключений до бази даних під керуванням MS SQL Server.

— Контролерів

— Хабів підключення технології SignalR.

Процес розгортання виконується при запуску серверу і відбувається один раз та паралельно для кожного сервісу. Після розгортання сервісів, в хаб SignalR очікує на підключення клієнтів, після чого відбувається перенаправлення на контролер авторизації клієнта. Після успішної авторизації клієнт отримує особливий токен, який підтверджує авторизацію клієнту і передається ним при виконані будь-яких дій. Процес авторизації детальніше розглянемо детальніше пізніше. У разі безуспішності авторизації підключення блокується на деякий час. Після одержання токену клієнт та сервер утримують підключення в умовах якого здійснюється передача повідомлень по двонаправленому каналу від клієнта до сервера та навпаки. Передача повідомлень здійснюється за допомого технології SignalR та відбувається в режимі реального часу. Зі сторони серверу є широкі можливості по конфігурації підключень.

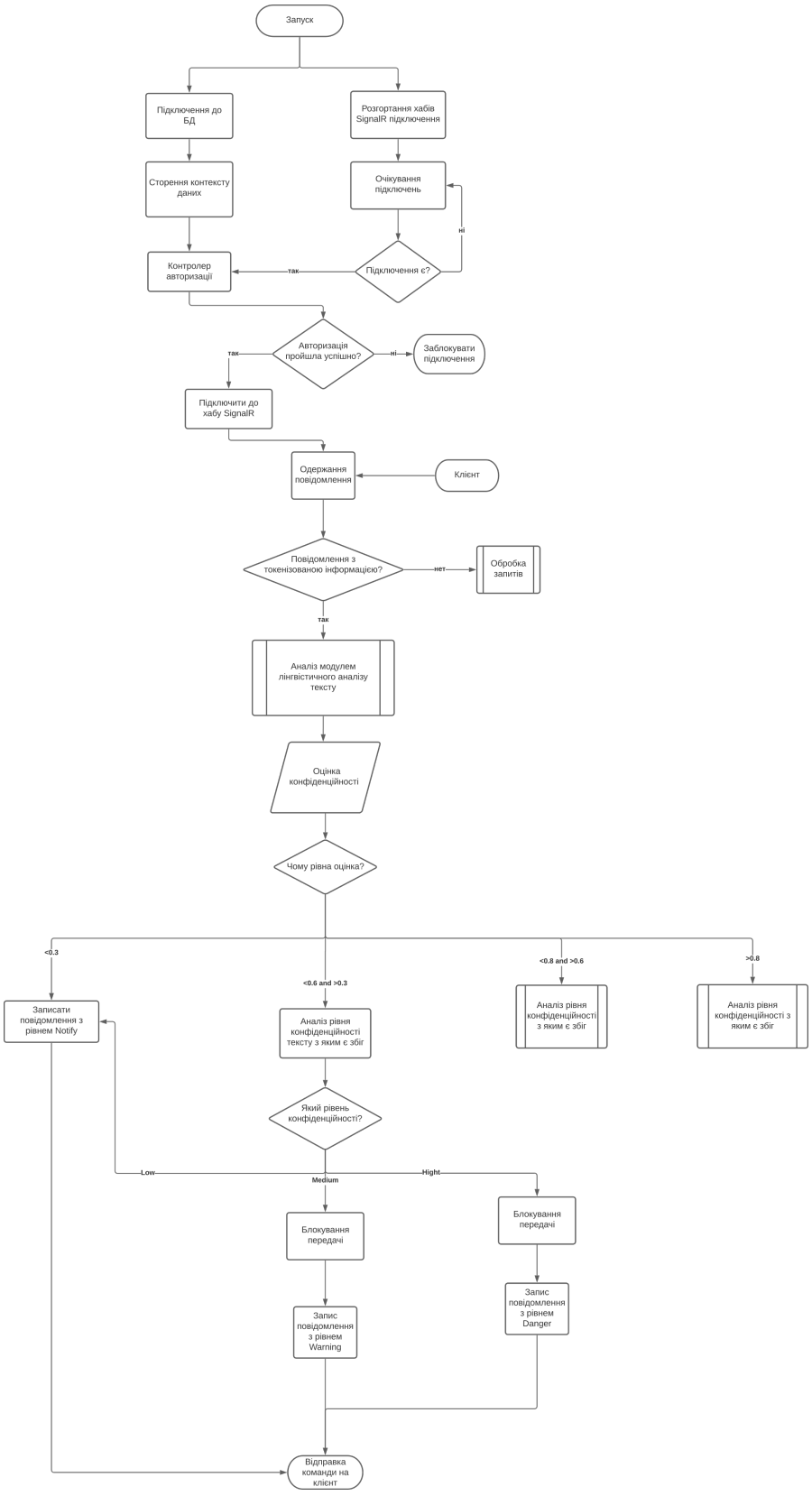



Рисунок 4.1 - Алгоритм обробки запитів перевірки на конфіденційність даних

Наприклад, час утримання активного підключень, в разі безактивності клієнту, встановлення схеми роботи схеми автентифікації та авторизації, час утримання «рукостискання» - підключення клієнта та сервера тощо.

У випадку отримання повідомлення від клієнту дія переходить до обробника події. У випадку отримання повідомлення із токенізованою інформацією для перевірки на предмет витоку, «відбиток» тексту відправляється на аналіз модулем лексичного аналізу, який оцінює рівень конфіденційності документа на основі подібності із іншими «відбитками» інформації, збереженими в базі даних системи захисту від витоку даних. В результаті отримуємо оцінку, на основі якої клієнту відсилається команда на блокування чи пропускання передачі інформації і також здійснюється запис повідомлення в базу даних із рівнем повідомлення Notify, Warning або Danger відповідно до оцінки та інформацією про клієнта, змістом повідомлення, оцінкою та датою й часом інциденту інформаційної безпеки.

Після виконання попередніх дій обробка повідомлення завершується і системи переходить до обробки наступного запису, або очікування повідомлення.

Переходимо до роботи алгоритму авторизації клієнтів на сервері системи захисту від витоків даних (рисунок 4.2). Ключовим аспектом реалізації авторизації в нашому алгоритмі є контролер шаблону програмування MVC та використання технології JWT Bearer. Контролер для виступає в ролі обробника запиту авторизації та викликає генератор JWT токенів, який в результаті успішної авторизації клієнта згенерує йому тимчасовий токен.

Відправною точку є звернення клієнта на контролер із запитом авторизації і передачею логіну та пароля. Наступним етапом є звернення до бази даних через контекст і отримання списку логінів та відповідному кожному логіну хешкодів.

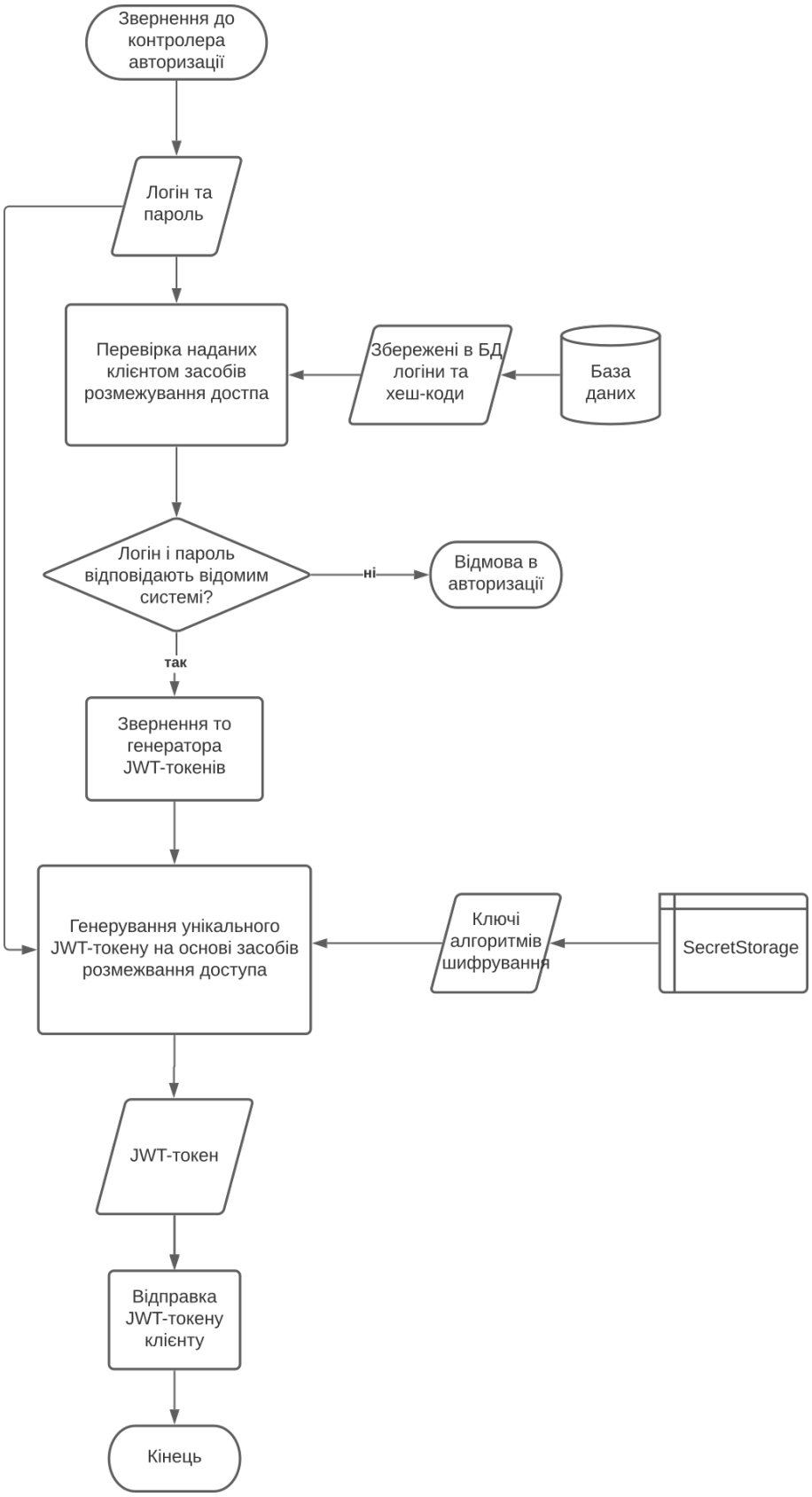



Рисунок 4.2 - Алгоритм авторизації клієнтів на сервері

Далі з допомогою ASP.Net Core Identity обчислюється хеш тексту та пароля отриманого від клієнту, і здійснюється перевірка із хешсумою збереженою в базі даних. У разі відповідності логіну та хешсумм клієнт проходить авторизації. Наступним етапом є звернення до сервісу генератора JWT токенів. Із спеціалізованого секретного сховища дістаються ключі шифрування для симетричного алгоритму шифрування, що використовується для видачі токенів. На основі отриманого ключа, логіну та тексту здійснюється хешування та шифрування симетричним ключем. В результаті ми отримуємо JWT токен, який може тимчасово підтверджувати клієнта. Результат відправляється клієнту.

Розглянемо алгоритм роботи головного компоненту системи – модуля лексичного аналізу тексту – зображеного на рисунку 4.3. Першим кроком є ініціалізацію компоненту як сервісу в середовищі ASP.Net Core. Наступним етапом є підготовка до отримання повідомлень та їх обробки. Відбувається два паралельних процеси. Перший з яких це створення черги аналізованих текстів, другий це процес обробки запитів ззовні модуля лексичного аналізу. При появі нових запитів на аналіз витоку даних здійснюється запис тексту і додаткової інформації про клієнта, який здійснив запит, в чергу аналізованого тексту.

Основною метою роботи черги є впорядкування запитів аналізованих текстів і оптимізація роботи модуля. Якщо в черзі є неопрацьований відбиток текст ініціалізується процес оцінювання рівня конфіденційності. Процес здійснюється двома методами:

— Метод простого порівняння. Виконується звичайним перебором відбитків текстів в збережених в базі даних системи, як конфіденційні, із відбитком аналізованого тексту.

— Метод «мішка слів». Реалізовується створенням матриці де рядки це лексеми(токен) аналізованого тексту, а стовпці це токен збереженого тексту. Рядками є лексеми, а стовпцями є аналізований відбиток тексту та збережені в системі відбитки. На перетині вказується кількість входжень лексеми в текстах.

В результаті ми отримуємо матрицю, яка описує вектори в багатовимірному просторі. Тепер ми приймаємо, що тексти є даними векторами і рахуємо скалярний добуток вектора аналізованого відбитку із ішими векторами попарно. В результаті ми отримаємо оцінку подібності відбитків і в той же час рівень конфіденційності аналізованого тексту. Наступним етапом є вибір оцінки на основі яка буде надана тексту. Для цього обирається текст із найбільшою оцінкою та найвищим рівнем конфіденційності.

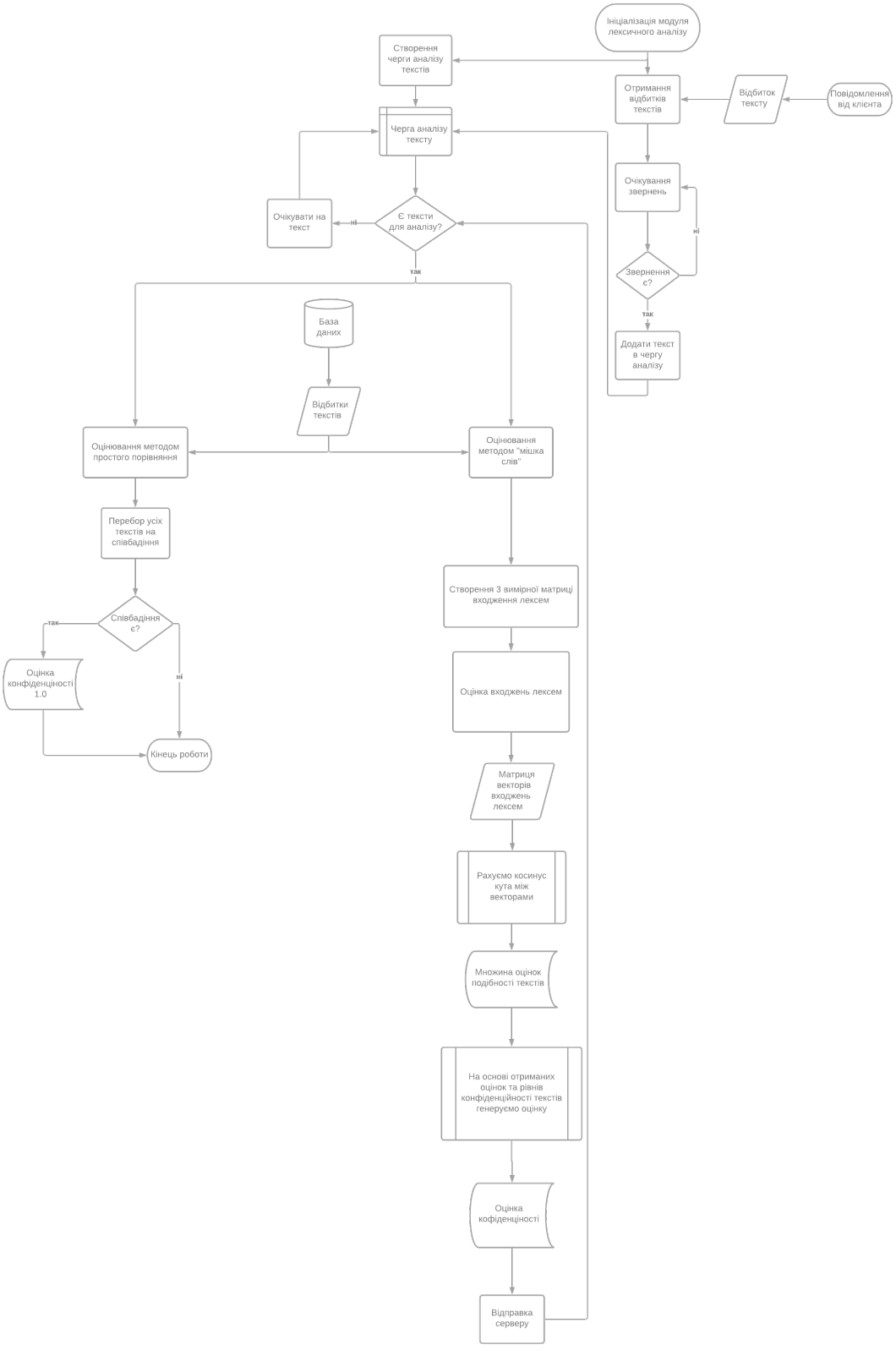



Рисунок 4.3 - Алгоритм роботи модуля лексичного аналізу тексту

## 4.2 Тестування системи

Для тестування системи було створено мережу із декількох віртуальних машин на яких було встановлено клієнт системи захисту від вити току даних, а також додаток який емалював діяльність роботи користувача, а саме здійснював операцію переміщення файлів та копіювання текстів і файлів. Сервером виступала віртуальна машина під керування Linux Ubuntu. На якій був розгорнутий контейнер docker в якому містився сервер керування системою захисту.

Додатки емалювання діяльності користувача мали спільне сховище із документами, які використовувалися для операцій копіювання/переміщення файлів.

На рисунку 4.4 зображений графік кількості запитів і кількість витоків які було дійсно скоєно. Я обав початковим значенням 1000 операцій на годину і поступовим зменшенням із кроком 100. Кількість витоків задавалася випадково оператором тестування.

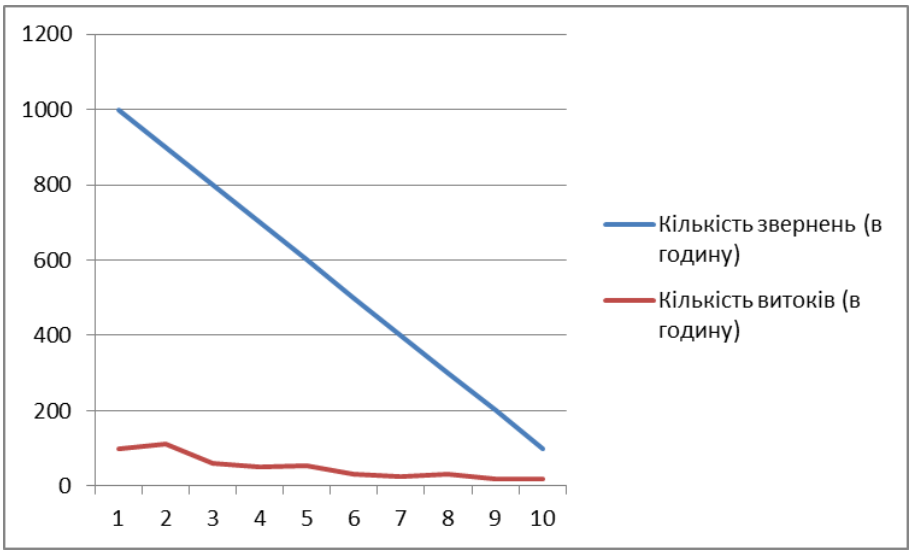



Рисунок 4.4 - Графік кількості запитів і кількості реальних витоків

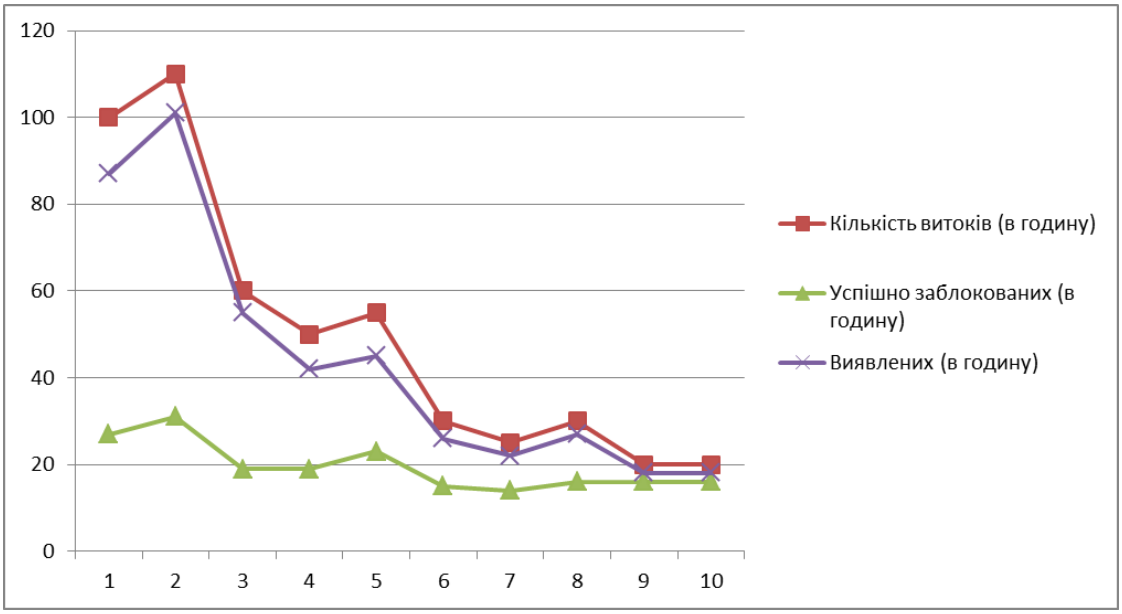



Рисунок 4.5 - Графік кількості успішно виявлених і заблокованих витоків

Як видно з рисунку 4.5 і 4.6 кількість успішних виявлень витоків доволі висока і становить близько 90%. Проте кількість успішних блокувань доволі критично залежить від кількості запитів. Причиною даної залежності є архітектура модуля лінгвістичного аналізу тексту, а саме рішення із організації аналізу тексту у вигляді черги запитів. За рахунок цього модуль не може швидко закінчити аналіз тексту до моменту здійснення операцій над файлом із конфіденційною інформацією. Також на ефективність вливає те, що тестування відбувається на одній реальній машині. Дане тестування вказало на проблеми даної архітектури. Особливої уваги потрібно надати потужностям серверу і архітектурі модуля лексичного аналізу, а саме підвищення ефективності методу оцінки і розпаралелювання процесу його здійснення.

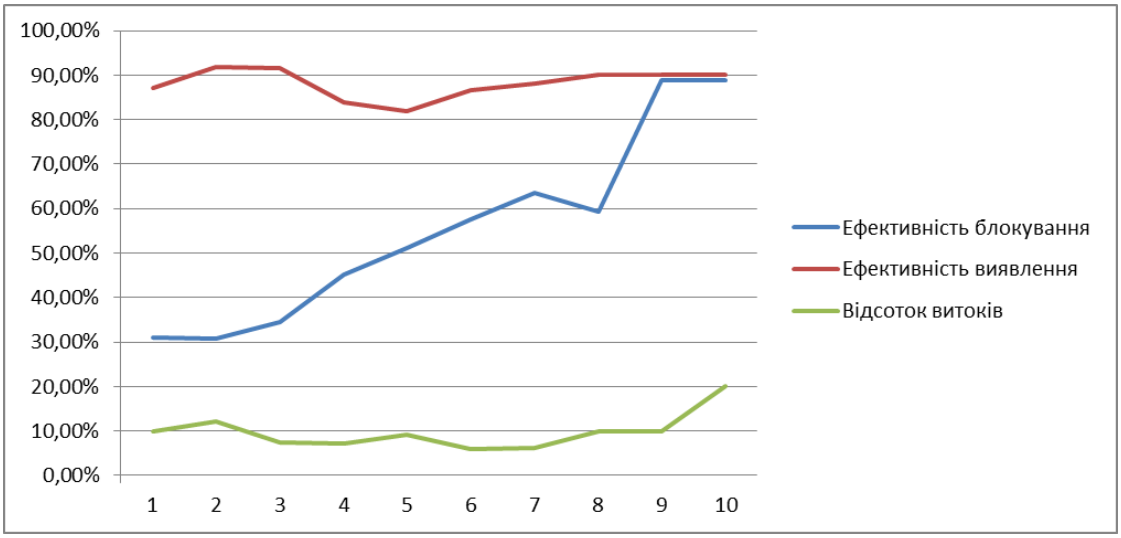



Рисунок 4.6 - Графік ефективності виявлення і блокування витоків даних

## 4.3 Впровадження системи в промислову експлуатацію

Так як система виконана у вигляді класичної клієнт-серверної архітектури, то для роботи системи необхідно виділений сервер із встановленою серверною частиною системи, а на робочі станції на яких потрібно відслідковувати витоки даних необхідно встановити клієнт.

Особливу увагу потрібно приділити потужностям роботи серверу, так як в пункті 4.2 була виявлена залежність ефективності роботи системи захисту від продуктивності сервера де розгорнутий серверний додаток.

Підтримка та адміністрування системи здійснюється за допомогою вебпанелі описаної в 3 розділі. Оновлення системи необхідно робити в разі потреби зміни функціоналу системи захисту або через значне оновлення технічної бази інформаційно-комунікаційної системи.

# ВИСНОВКИ

Дана кваліфікаційна робота вказує на важливість і перспективність розробок пов’язаних із забезпеченням безпеки в кібернетичному просторі. Така тенденція пов’язана із бурхливим розвитком і впровадженням інформаційних технологій у життя сучасного суспільства. В роботі описані традиційні проблеми інформаційної безпеки і кібернетичному просторі і заходи пов’язані із забезпечення захисту від загроз інформаційної безпеки. Особливу увагу було зосереджено на проблемі витоків даних із конфіденційною інформацією причиною якого є внутрішній порушник інформаційної безпеки, тобто загрози зсередини системи.

Як показує статистика, причиною інцидентів інформаційної безпеки кількість витоків конфіденційною інформації дедалі більше стають співробітники організацій та авторизовані в середині системи, а не зовнішні порушники. Дана тенденція викликає необхідність до переосмислення підходів до забезпечення інформаційної безпеки і створення нових розробок із забезпеченням безпеки не лише від зовнішніх порушників, але й від внутрішніх.

В рамках роботи була спроектована системи захисту від витоків даних, яка націлена на забезпечення потреб превентивного захисту інформації у Відділенні протидії кіберзлочинам Департаменту кіберполіції Національної поліції України в Хмельницькій області. Система була спроектована враховуючи потреби та побажання співробітників. Для цього було розроблене технічне завдання на систему, згідно якого в подальшому виконувалася проектна розробка і реалізація системи мовою програмування C# використовуючи переваги платформи ASP.Net Core для адміністрування системи, а також технології передачі повідомлень в режимі реального часу SignalR.

В результаті тестування розробленої системи захисту від витоків даних було виявлено недолік, пов’язаний із стійкістю до навантаження системи. Проте даний недолік не є критичним через локальну сферу застосування розробленої системи. Але це може стати підґрунтям для подальшої наукової розробки пов’язаною із розробкою унікального методу аналізу витоку даних, який може застосуватися в середині даної системи захисту і значно покращити ефективність розпізнавання конфіденційної інформації серед масиву даних, циркулюючих в середині інформаційно-комунікаційного середовища.

## ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ


1) Curtis Waltman. Aurora: Homeland Security's secret project to change how we think about cybersecurity. [Електронний ресурс]. – Режим доступа: https://www.muckrock.com/news/archives/2016/nov/14/aurora-generator-test-homeland-security/ (дата звернення 10.05.2021). – Назва з екрану

2) Cyber Security for Cyber Physical Systems / Saqib Ali, Taiseera Al Balushi, Zia Nadir, Omar Khadeer Hussain. – Cham, Switzerland : Springer, 2018. – 174 p.

3) ISO/IEC 27000:2018(e) international standard. Information technology — security techniques — information security management systems — overview and vocabulary — fifth edition 2018-02.

4) Бурячок В. Л. Основи інформаційної та кібернетичної безпеки: навчальний посібник / В. Л. Бурячок, Р. В. Киричок, П. М. Складанний – К., 2018. – 320 с.

5) Закон України «Про інформацію» (Відомості Верховної Ради України (ВВР), 1992, № 48, ст.650) [Електронний ресурс] / Веб-портал Верховної ради України. – Режим доступу: http://zakon2.rada.gov.ua/laws/ – Назва з екрана.

6) Інформаційна безпека держави: навчальний посібник/ В.І. Гур'єв, Д.Б. Мехед, Ю.М. Ткач, І.В. Фірсова. – Ніжин: ФОП Лук'яненко В.В. ТПК «Орхідея», 2018. – 166 с.

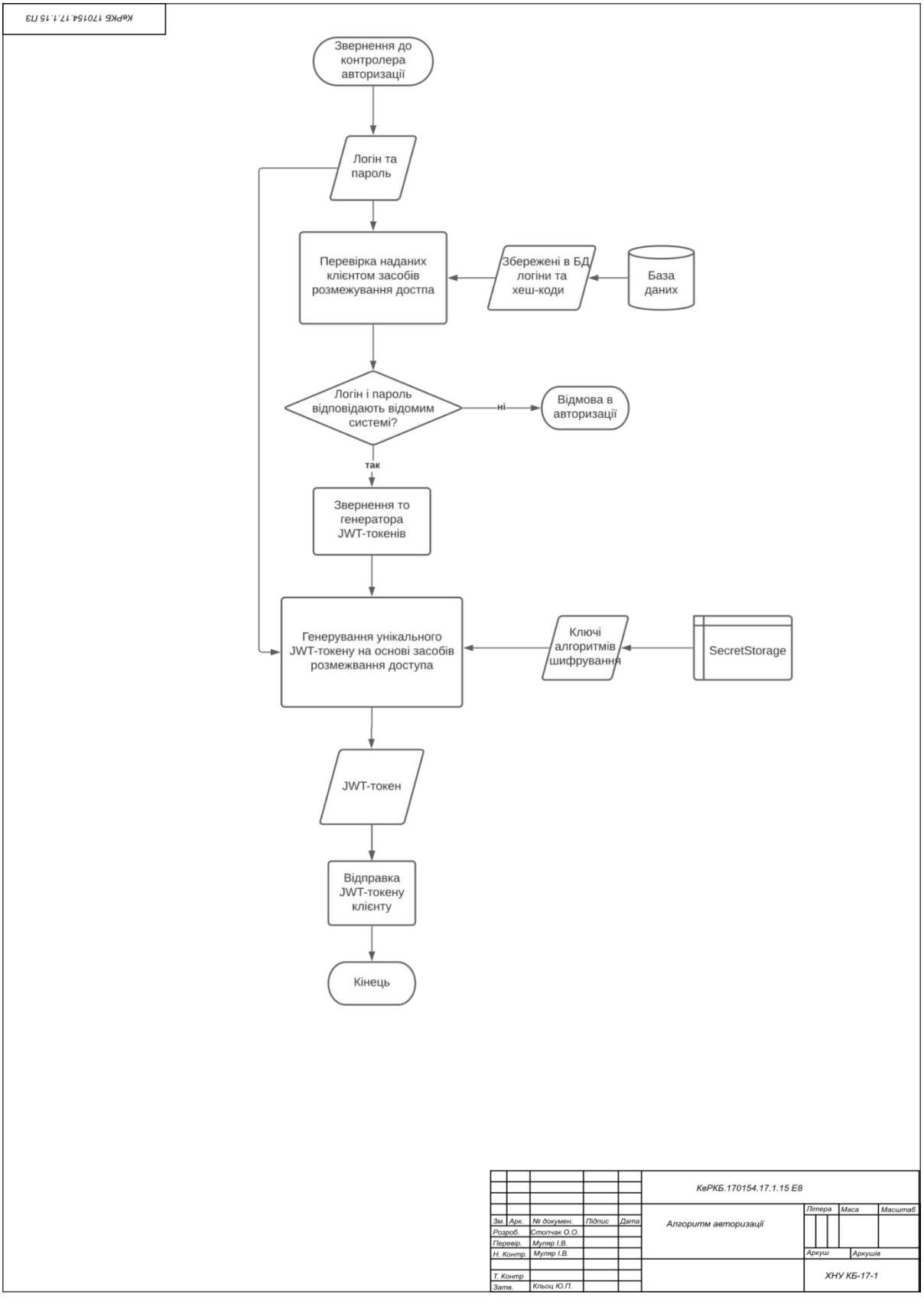
7) Інформаційна безпека: навчальний посібник / [Ю. Я. Бобало, І. В. Горбатий, М. Д. Кіселичник, А. П. Бондарєв та ін.]; за заг. ред. д-ра техн. наук, проф. Ю. Я. Бобала та д-ра техн. наук, доц. І. В. Горбатого. – Львів: Видавництво Львівської політехніки, 2019. – 580 с.

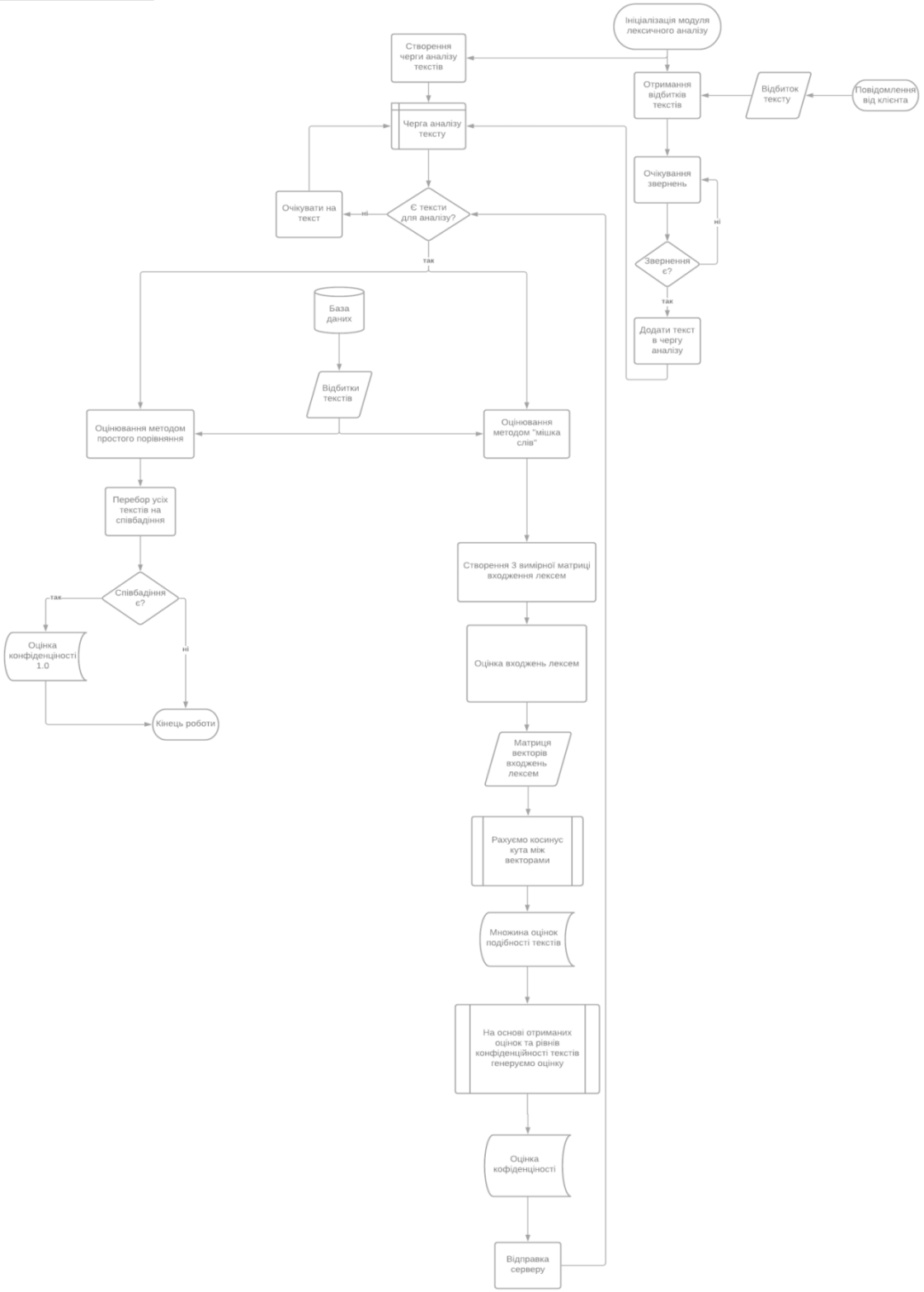
8) Кваліфікаційна робота бакалавра . Методичні вказівки щодо її виконання для студентів спеціальності 125 "Кібербезпека" освітнього

# ДОДАТОК А

(Обов’язковий)

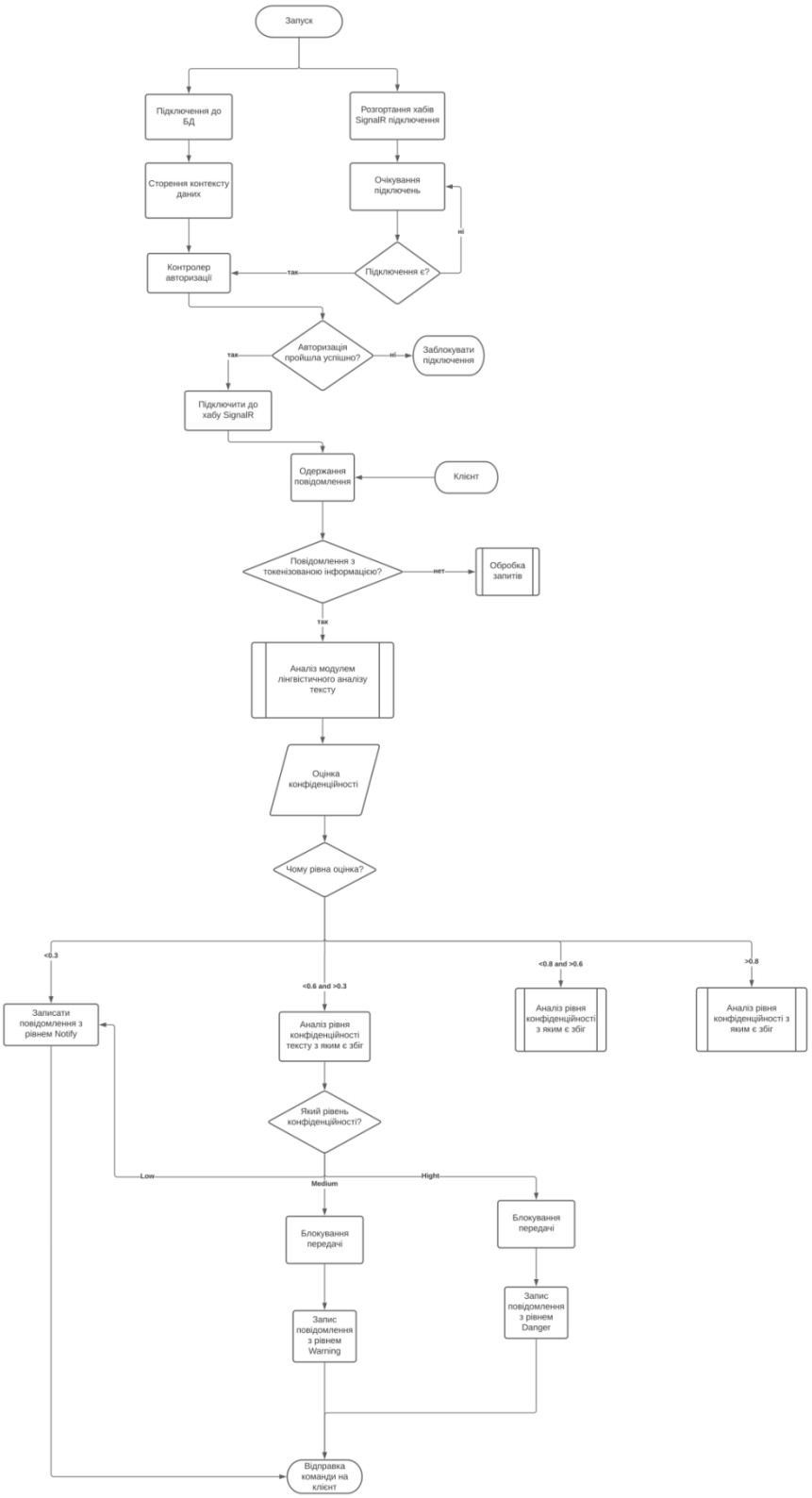
## Копія графічної частини





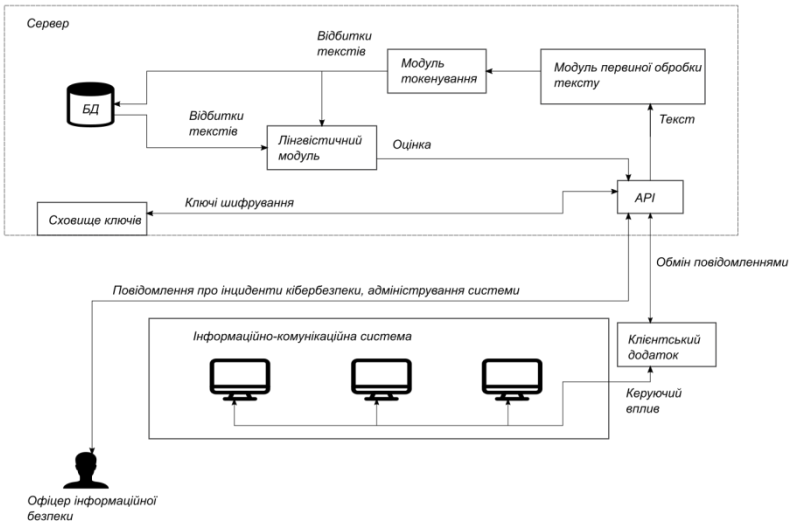

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | КвРКБ.170154.17.1.15 Е8 | |
| Зм. | Арк. | № докумен. | Підпис | Дата | Алгоритм роботи модуля лексичного аналізу | Літера / Маса / Масштаб |
| Розроб. | | Стопчак О.О. | | | | |
| Перевір. | | Муляр І.В. | | | | Аркуш / Аркушів |
| Н. Контр. | | Муляр І.В. | | | | |
| Т. Контр | | | | | ХНУ КБ-17-1 | |
| Затв. | | Кльоц Ю.П. | | | | |

КвРКБ 170154.17.1.15 ПЗ

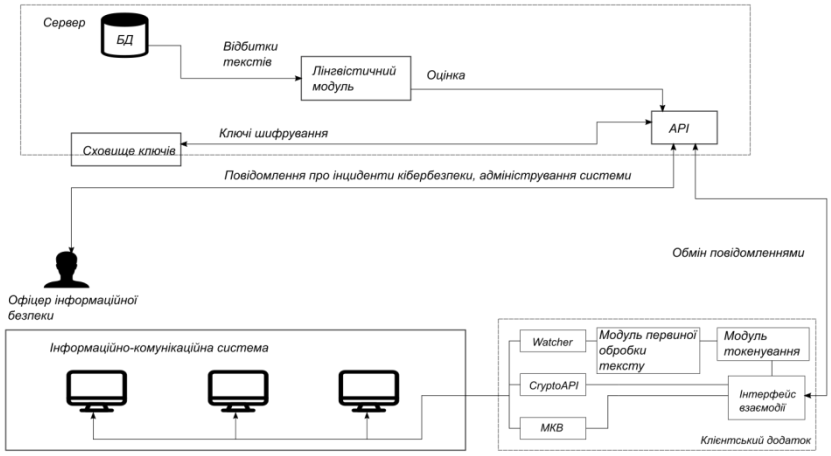



| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | КвРКБ.170154.17.1.15 Е8 | |
| Зм. | Арк. | № докумен. | Підпис | Дата | Алгоритм роботи сервера | Літера / Маса / Масштаб |
| Розроб. | | Стопчак О.О. | | | | |
| Перевір. | | Мупяр І.В. | | | | Аркуш / Аркушів |
| Н. Контр. | | Мупяр І.В. | | | | |
| Т. Контр | | | | | ХНУ КБ-17-1 | |
| Затв. | | Кльоц Ю.П. | | | | |

КвРКБ 170154.17.1.15 ПЗ




| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | КвРКБ.170154.17.1.15 ПЗ | |
| | | | | | | Літера / Маса / Масштаб |
| Зм. | Арк. | № докумен. | Підпис | Дата | Архітектура системи захисту від витоків даних | |
| Розроб. | | Сточчак О.О. | | | | |
| Перевір. | | Мулер І.В. | | | | Аркуш / Аркушів |
| Н. Контр. | | Мулер І.В. | | | | |
| Т. Контр | | | | | | ХНУ КБ-17-1 |
| Затв. | | Кільоц Ю.Л. | | | | |




| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | КвРКБ.170154.17.1.15 ПЗ | |
| | | | | | | | |
| Зм. | Арк. | № докумен. | Підпис | Дата | | | |
| Розроб. | | Сточчак О.О. | | | Архітектура системи захисту від витоків даних модифікована | Літера | Маса / Масштаб |
| Перевір. | | Мулер І.В. | | | | | |
| Н. Контр. | | Мулер І.В. | | | | Аркуш | Аркушів |
| | | | | | | | |
| Т. Контр | | | | | | ХНУ КБ-17-1 | |
| Затв. | | Кльоц Ю.П. | | | | | |

# ДОДАТОК Б

## (Обов'язковий)

## Програмна реалізація

```
using Microsoft.AspNetCore.Identity.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore;
using System;
using System.Linq;
using System.Threading.Tasks;

namespace ServerSignalR.Models
{
   public class ApplicationContext : IdentityDbContext<User, Role, Guid>
   {
      public DbSet<Notify> Notifies { get; set; }
      public DbSet<Text> Texts { get; set; }
      public DbSet<Token> Tokens { get; set; }

      public ApplicationContext(DbContextOptions<ApplicationContext> options)
         : base(options)
      {
         Database.EnsureCreated();
      }
      protected override void OnModelCreating(ModelBuilder modelBuilder)
      {
         modelBuilder.Entity<User>().Property(p => p.Id).ValueGeneratedOnAdd();
         base.OnModelCreating(modelBuilder);
      }
   }
}


using Microsoft.AspNetCore.Identity;
using System;

namespace ServerSignalR.Models
{
   public class User:IdentityUser<Guid>
   {

   }
}
using System;
using System.Collections.Generic;

namespace ServerSignalR.Models
{
   public class Token
   {
      public Guid Id { get; set; }
      public string Value { get; set; }
```

```
        public List<Text> Texts { get; set; }
    }
}
using System;
using System.Collections.Generic;

namespace ServerSignalR.Models
{
    public class Text
    {
        public Guid Id { get; set; }
        public string Name { get; set; }
        public LevelConfidence LevelConfidence { get; set; }
        public Guid UserId { get; set; }
        public User User { get; set; }
        public List<Token > Tokens { get; set; }
    }
}
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;

namespace ServerSignalR.Models
{
    public class Notify
    {
        public Guid Id { get; set; }
        public DateTime DateTime { get; set; }
        public string Message { get; set; }
        public Guid UserId { get; set; }
        public User User { get; set; }
        public NotifyType NotifyType { get; set; }
    }
}
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using Microsoft.IdentityModel.Tokens;
using ServerSignalR.JWT;
using ServerSignalR.Models;
using System;
using System.Collections.Generic;
using System.IdentityModel.Tokens.Jwt;
using System.Security.Claims;
using System.Threading.Tasks;

namespace ServerSignalR.Controllers
{
    public class AccountController : Controller
    {
        private ApplicationContext _context;
        public AccountController(ApplicationContext context)
        {
            _context = context;
        }
        [HttpPost("/token")]
        public async Task<IActionResult> Token(string username, string password)
```

```
        {
            var identity = await GetIdentity(username, password);
            if (identity == null)
            {
                return BadRequest("Invalid username or password.");
            }

            var now = DateTime.UtcNow;
            var jwt = new JwtSecurityToken(
                    issuer: AuthOptions.ISSUER,
                    audience: AuthOptions.AUDIENCE,
                    notBefore: now,
                    claims: identity.Claims,
                    expires: now.Add(TimeSpan.FromMinutes(AuthOptions.LIFETIME)),
                    signingCredentials: new SigningCredentials(AuthOptions.GetSymmetricSecurityKey(),
SecurityAlgorithms.HmacSha256));
            var encodedJwt = new JwtSecurityTokenHandler().WriteToken(jwt);

            var response = new
            {
                access_token = encodedJwt,
                username = identity.Name
            };
            return Json(response);
        }

        private async Task<ClaimsIdentity> GetIdentity(string username, string password)
        {
            User person = await _context.Users.Include(r => r.Role).FirstOrDefaultAsync(x => x.Email ==
username && x.Password == password);
            if (person != null)
            {
                var claims = new List<Claim>
                {
                    new Claim(ClaimsIdentity.DefaultNameClaimType, person.Email),
                    new Claim(ClaimsIdentity.DefaultRoleClaimType, person.Role.Name)
                };
                ClaimsIdentity claimsIdentity =
                new ClaimsIdentity(claims, "Token", ClaimsIdentity.DefaultNameClaimType,
                    ClaimsIdentity.DefaultRoleClaimType);
                return claimsIdentity;
            }
            return null;
        }
    }
}
using Microsoft.AspNetCore.Authentication.JwtBearer;
using Microsoft.AspNetCore.Builder;
using Microsoft.AspNetCore.Hosting;
using Microsoft.AspNetCore.HttpsPolicy;
using Microsoft.EntityFrameworkCore;
using Microsoft.Extensions.Configuration;
using Microsoft.Extensions.DependencyInjection;
using Microsoft.Extensions.Hosting;
using Microsoft.IdentityModel.Tokens;
using ServerSignalR.JWT;
using ServerSignalR.Models;
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;

namespace ServerSignalR
{
   public class Startup
   {
      public Startup(IConfiguration configuration)
      {
         Configuration = configuration;
      }

      public IConfiguration Configuration { get; }

      // This method gets called by the runtime. Use this method to add services to the container.
      public void ConfigureServices(IServiceCollection services)
      {
         string connection =
"Server=(localdb)\\mssqllocaldb;Database=DiplomDB;Trusted_Connection=True;";
         services.AddDbContext<ApplicationContext>(options => options.UseSqlServer(connection));

         services.AddAuthentication(JwtBearerDefaults.AuthenticationScheme)
            .AddJwtBearer(options =>
            {
               options.RequireHttpsMetadata = false;
               options.TokenValidationParameters = new TokenValidationParameters
               {
                  ValidateIssuer = true,
                  ValidIssuer = AuthOptions.ISSUER,
                  ValidateAudience = true,
                  ValidAudience = AuthOptions.AUDIENCE,
                  ValidateLifetime = true,
                  IssuerSigningKey = AuthOptions.GetSymmetricSecurityKey(),
                  ValidateIssuerSigningKey = true,
               };
               options.Events = new JwtBearerEvents
               {
                  OnMessageReceived = context =>
                  {
                     var accessToken = context.Request.Query["access_token"];
                     var path = context.HttpContext.Request.Path;
                     if (!string.IsNullOrEmpty(accessToken) &&
                        (path.StartsWithSegments("/chat")))
                     {
                        context.Token = accessToken;
                     }
                     return Task.CompletedTask;
                  }
               };
            });
         services.AddSignalR();
         services.AddControllers();
      }

      public void Configure(IApplicationBuilder app)
```

```
        {
            app.UseDeveloperExceptionPage();

            app.UseHttpsRedirection();
            app.UseDefaultFiles();
            app.UseStaticFiles();

            app.UseRouting();

            app.UseAuthentication();
            app.UseAuthorization();

            app.UseEndpoints(endpoints =>
            {
                endpoints.MapControllers();
                endpoints.MapHub<MessageHub>("/message");
            });
        }
    }
}
using System;
using System.IO;
using System.Security.Permissions;
using System.Threading.Tasks;

namespace Watcher.FileManagment
{
    public class FileManager
    {
        public delegate void FileManagerNotifyHandler(string message);
        public event FileManagerNotifyHandler Notify;

        public async Task RunAsync(string path)
        {
            await Task.Run(() => Run(path));
        }
        private void Run( string path)
        {
            // If a directory is not specified, exit program.
            if (!string.IsNullOrEmpty(path))
            {
                // Display the proper way to call the program.
                Notify("Usage: Watcher.exe (directory)");
                return;
            }

            // Create a new FileSystemWatcher and set its properties.
            using (FileSystemWatcher watcher = new FileSystemWatcher())
            {
                watcher.Path = path;
                watcher.IncludeSubdirectories = true;

                // Watch for changes in LastAccess and LastWrite times, and
                // the renaming of files or directories.
                watcher.NotifyFilter = NotifyFilters.LastAccess
                                     | NotifyFilters.LastWrite
                                     | NotifyFilters.FileName
```

```
                                | NotifyFilters.DirectoryName;

            // Only watch text files.
            watcher.Filter = "*.*";

            // Add event handlers.
            watcher.Changed += OnChanged;
            watcher.Created += OnChanged;
            watcher.Deleted += OnChanged;
            watcher.Renamed += OnRenamed;

            // Begin watching.
            watcher.EnableRaisingEvents = true;

            // Wait for the user to quit the program.
            //Notify("Press 'q' to quit the sample.");
            while (true) ;

        }
    }

    // Define the event handlers.
    private void OnChanged(object source, FileSystemEventArgs e) =>
        // Specify what is done when a file is changed, created, or deleted.
        Notify($"File: {e.FullPath} {e.ChangeType}");

    private void OnRenamed(object source, RenamedEventArgs e) =>
        // Specify what is done when a file is renamed.
        Notify($"File: {e.OldFullPath} renamed to {e.FullPath}");
  }
}
using System.Collections.Generic;
using Watcher.LexicalAnalyze.Params;

namespace Watcher.LexicalAnalyze.Core
{
  public interface ILexicalData
  {
      List<Text> Texts { get; set; }
      List<Lexam> Lexams { get; set; }
  }
}using Watcher.LexicalAnalyze.Params;

namespace Watcher.LexicalAnalyze.Core
{
  public interface ILexicalSearch
  {
    public LevelOfSimilarity Search(string lexes);
  }
}
namespace Watcher.LexicalAnalyze.Params
{
  public enum LevelOfSimilarity
  {
    UltraLow,
    Low,
    Medium,
```

```
        Hight,
        UltraHight,
        Similyar
    }
}
namespace Watcher.LexicalAnalyze.Params
{
    public class Lexam
    {
        public int Id { get; set; }
        public string Lex { get; set; }
        public string NormolizedLex { get; set; }
        public int Value { get; set; }
        public static int ID { get; set; } = 0;
        public Lexam()
        {
            Id = ++ID;
            Value = Id;
        }
    }
}
using System;

namespace Watcher.LexicalAnalyze.Params
{
    public class Text
    {
        public int Id { get; set; }
        public string TextName { get; set; }
        public string Tokens{ get; set; }
        public TextPrivacy Privacy { get; set; }
        public DateTime DateAdd { get; set; }
        public static int ID;
        public Text()
        {
            Id = ++ID;
            DateAdd = DateTime.Now;
        }

    }
}
namespace Watcher.LexicalAnalyze.Params
{
    public enum TextPrivacy
    {
        Low,
        Medium,
        Hight
    }
}using DocumentFormat.OpenXml.Packaging;
using DocumentFormat.OpenXml.Wordprocessing;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
```

```
namespace Watcher.LexicalAnalyze.TextRead
{
   public class DocReader : IFileRead
   {
      public async Task<string> Read(string path)
      {
         WordprocessingDocument wordprocessingDocument =
     WordprocessingDocument.Open(path, true);
         Body body = wordprocessingDocument.MainDocumentPart.Document.Body;
         string result = "";
         foreach (var t in body)
         {
            if (!string.IsNullOrEmpty(t.InnerText))
               result += t.InnerText+" ";
         }
         wordprocessingDocument.Close();

         return await Task.FromResult(result);
      }
   }
}
using System.Threading.Tasks;

namespace Watcher.LexicalAnalyze.TextRead
{
   public interface IFileRead
   {
      public Task<string> Read(string path);
   }
}
using iTextSharp.text.pdf;
using iTextSharp.text.pdf.parser;
using System.Text;
using System.Threading.Tasks;

namespace Watcher.LexicalAnalyze.TextRead
{
   public class PDFReader : IFileRead
   {
      public async Task<string> Read(string path)
      {
         StringBuilder text = new StringBuilder();


         PdfReader pdfReader = new PdfReader(path);

         for (int page = 1; page <= pdfReader.NumberOfPages; page++)
         {
            ITextExtractionStrategy strategy = new SimpleTextExtractionStrategy();
            string currentText = PdfTextExtractor.GetTextFromPage(pdfReader, page, strategy);

            currentText = Encoding.UTF8.GetString(ASCIIEncoding.Convert(Encoding.Default,
Encoding.UTF8, Encoding.Default.GetBytes(currentText)));
            text.Append(currentText);
         }
         pdfReader.Close();
```

```
            return await Task.FromResult(text.ToString());
        }
    }
}
using System.IO;
using System.Threading.Tasks;

namespace Watcher.LexicalAnalyze.TextRead
{
    public class TxtReader : IFileRead
    {
        public async Task<string> Read(string path)
        {

            using StreamReader sr = new StreamReader(path);
            string result = await sr.ReadToEndAsync();
            return result;
        }
    }
}
using System.Collections.Generic;
using Watcher.LexicalAnalyze.Core;
using Watcher.LexicalAnalyze.Params;

namespace Watcher.LexicalAnalyze
{
    public class LexicalData : ILexicalData
    {
        public List<Text> Texts { get; set; } = new List<Text>();
        public List<Lexam> Lexams { get; set; } = new List<Lexam>();
    }
}
using System.IO;
using System.Linq;
using System.Threading.Tasks;
using Watcher.LexicalAnalyze.Core;
using Watcher.LexicalAnalyze.Params;
using Watcher.LexicalAnalyze.TextRead;

namespace Watcher.LexicalAnalyze
{
    public class LexicalAnalyzer
    {
        private ILexicalData lexicalData;
        private ILexicalSearch lexicalSearch;
        public LexicalAnalyzer(ILexicalData lexicalData, ILexicalSearch lexicalSearch)
        {
            this.lexicalSearch = lexicalSearch;
            this.lexicalData = lexicalData;
        }
        public LexicalAnalyzer(ILexicalData lexicalData) : this(lexicalData, new
WordBagsLexicalSearch(lexicalData))
        {

        }
```

```
        private string[] Delimiter = { ".", ";", ",", "(", ")", "+", "-", "*", "/", "=", ">",
"<","\\","[","]","\r","\n"," " };

        public async Task<string> ReadFileAsync(string path)
        {
            if (!File.Exists(path))
            {
                throw new FileNotFoundException();
            }
            switch (path.Split(".").Last())
            {
                case "txt":
                    return await new TxtReader().Read(path);
                case "doc":
                case "docx":
                    return await new DocReader().Read(path);
                case "pdf":
                    return await new PDFReader().Read(path);
                default:
                    throw new FileFormatException("File format is not suported");
            }
        }
        public async Task AddTextAsync(string textName, string path, TextPrivacy textPrivacy =
TextPrivacy.Low)
        {
            string buffer = await ReadFileAsync(path);
            Text text = new Text();
            text.Privacy = textPrivacy;
            text.TextName = textName;
            text.Tokens = Tokenization(buffer);
            lexicalData.Texts.Add(text);
        }
        private string Tokenization(string text)
        {
            string result="";
            string[] textAfterSplit = text.Split(Delimiter,System.StringSplitOptions.RemoveEmptyEntries);
            foreach (var t in textAfterSplit)
            {
                int indexOfLexam = lexicalData.Lexams.FindIndex(p => p.NormolizedLex == t.ToUpper());
                if (indexOfLexam==-1)
                {
                    var lexam = new Lexam { Lex = t, NormolizedLex = t.ToUpper() };
                    lexicalData.Lexams.Add(lexam);
                    result += lexam.Value + " ";
                }
                else
                {
                    result += lexicalData.Lexams[indexOfLexam].Value + " ";
                }

            }

            return result;
        }
        public async Task<LevelOfSimilarity> GetSimiliarityScoreAsync(string path)
        {
            string buffer =await ReadFileAsync(path);
```

```
            buffer = Tokenization(buffer);
            return lexicalSearch.Search(buffer);
        }

    }
}
using System;
using System.Collections.Generic;
using System.Linq;
using Watcher.LexicalAnalyze.Core;
using Watcher.LexicalAnalyze.Params;

namespace Watcher.LexicalAnalyze
{
    public class WordBagsLexicalSearch : ILexicalSearch
    {
        private readonly List<Lexam> Lexams;
        private readonly List<Text> Texts;
        public WordBagsLexicalSearch(ILexicalData lexicalData)
        {
            Lexams = lexicalData.Lexams;
            Texts = lexicalData.Texts;
        }
        public LevelOfSimilarity Search(string lexes)
        {
            float maxMark = 0, bufMark;
            List<string> verifiableWordBug = GetWordBug(lexes);
            foreach(var t in Texts)
            {
                bufMark = getScoreOfSimilyarity(verifiableWordBug, GetWordBug(t.Tokens));
                if (bufMark > maxMark)
                    maxMark = bufMark;
            }
            if (maxMark == 1.0)
                return LevelOfSimilarity.Similyar;
            else if (maxMark >= 0.9)
                return LevelOfSimilarity.UltraHight;
            else if (maxMark >= 0.8)
                return LevelOfSimilarity.Hight;
            else if (maxMark >= 0.65)
                return LevelOfSimilarity.Medium;
            else if (maxMark >= 0.35)
                return LevelOfSimilarity.Low;
            else
                return LevelOfSimilarity.UltraLow;
        }
        private float getScoreOfSimilyarity(List<string> verifiableWordBug,List<string> originalWordBug)
        {
            int score = 0;
            if (verifiableWordBug.Count <= originalWordBug.Count)
            {
                foreach (var t in verifiableWordBug)
                    if (originalWordBug.Any(predicate => predicate == t))
                        score++;
                return (float)score / (float)verifiableWordBug.Count;
            }
            else
```

```
            {
                foreach (var t in originalWordBug)
                {
                    if (verifiableWordBug.Any(predicate => predicate == t))
                        score++;
                }
                return (float)score / (float)originalWordBug.Count;
            }
        }
        private List<string> GetWordBug(string text)
        {
            List<string> result = new List<string>();
            foreach(var t in text.Split(" ",StringSplitOptions.RemoveEmptyEntries))
            {
                if (result.IndexOf(t) == -1)
                {
                    result.Add(t);
                }
                else continue;
            }
            return result;

        }
    }
}
```

# РІШЕННЯ ЕКСПЕРНОЇ КОМІСІЇ
# КАФЕДРИ КІБЕРБЕЗПЕКИ ТА КОМП'ЮТЕРНИХ СИСТЕМ І МЕРЕЖ
# ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатом звіту подібності щодо роботи, генерованого системою виявлення текстових збігів/ідентичності/схожості:

Назва: Система захисту від витоку даних Відділу протидії кіберзлочинам Департаменту кіберполіції Національної поліції України

Автор: Стопчак Олександр Олександрович

Спеціальність: 125 – Кібербезпека

Освітня програма: Кібербезпека

Керівник: Муляр Ігор Володимирович, к.т.н., доцент

Після аналізу звіту подібності зроблено такий висновок:

| № | Висновок | Позначка про відповідність |
|---|---|---|
| 1 | Запозичення, виявлені в роботі, є законними і не є плагіатом (далі – зазначаються підстави віднесення запозичень до правомірних). Робота приймається до захисту. | відповідає |
| 2 | Виявлені запозичення не є плагіатом, розмішені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи (далі – зазначаються детальні та аргументовані підстави віднесення запозичень до правомірних). Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої та електронної версії роботи. | |
| 3 | Виявлені запозичення не є плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту (наступного року) після того як буде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат. | |
| 4 | Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту. | |
| 5 | Інше: | |

Підтвердження:

Запозичення, виявлені в роботі, є законними і не є плагіатом, оскільки:

1) запозичення розміщені в розділах аналізу існуючих аналогів та прототипів, які не описують безпосередньо авторське дослідження і не стосуються результатів роботи;
2) усі запозичення фрагментарні, або мають належним чином оформленні посилання;
3) всі зафіксовані системою ознаки модифікації тексту відносяться до комбінування латинських символів зі україномовними скороченнями індексів в формулах, що не є модифікацією тексту.

Сумарний обсяг всіх запозичень, визначений системою виявлення збігів/ ідентичності/схожості, складає 0.9% що, з урахуванням наведених обгрунтувань, відповідає характеру наукового дослідження і свідчить на користь кваліфікаційної роботи.

Керівник роботи _____ І.В. Муляр

Завідувач кафедри КбКСМ, гарант ОП _____ Ю.П. Кльоц

Дата: 07.06.2021

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

# РЕЦЕНЗІЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

освітнього ступеня «бакалавр»

Студент Стопчак Олександр Олександрович

Тема Система захисту від витоку даних Відділу протидії кіберзлочинам Департаменту кіберполіції Національної поліції України

Спеціальність 125 – Кібербезпека

**Обсяг кваліфікаційної роботи освітньо-кваліфікаційного рівня «бакалавр»:**

кількість листів креслень 5; кількість сторінок записки 58.

1. Короткий зміст роботи та прийнятих рішень У кваліфікаційній роботі розроблено DLP систему для захисту від витоку персональних даних.

2. Висновок про відповідність кваліфікаційної роботи завданню Кваліфікаційна робота у повній мірі відповідає поставленому завданню як в теоретичній, так і в практичній частині

3. Характеристика виконання кожного розділу роботи, ступінь використання останніх досягнень науки і техніки і передових методів роботи: У вступі подана загальна характеристика поставленої задачі, чітко визначено об'єкт, предмет та методи дослідження, сформульована актуальність. Визначені задачі, які необхідно вирішити для досягнення поставленої мети, практична цінність отриманих результатів, їхня новизна та наведені відомості про публікації. У першому розділі проведено огляд використовуваних в комп'ютерних системах методів захисту конфіденційної інформації, виконане обгрунтування актуальності теми дослідження і виконана постановка задачі. В другому розділі наведені засоби та технології використані для побудови системи захисту. В третьому розділі визначено основні положення системи та розроблено алгоритми її роботи. Четвертий розділ було присвячено апробації системи захисту та алгоритмів її реалізації.

4. Позитивні сторони роботи Кваліфікаційна робота має комплексну практичну цінність. Практична цінність полягає у розробці модуля лексичного аналізу з допомогою якого визначається ступінь конфіденційності даних. На основі даного аналізу в подальшому здійснюється керуючий вплив на витік конфіденційних даних. Практична цінність результатів кваліфікаційної роботи полягає у створені системи захисту від витоків даних, що може бути підлаштована для будь якої категорії даних, і у описі оптимальних моделей функціонування систем захисту подібного характеру.

5. Негативні сторони роботи Розроблена система захисту від витоків даних досить чутлива до навантаження.

6. Оцінка графічного оформлення та пояснювальної записки роботи Графічне оформлення виконане відповідно до теми кваліфікаційної роботи з дотриманням стандартів. В загальному графічне оформлення виконане якісно, пояснювальна записка відповідає нормам щодо її оформлення.

7. Відгук про роботу в цілому В загальному кваліфікаційна робота заслуговує позитивної оцінки. Весь матеріал кваліфікаційної роботи структурований, чіткий та послідовний. Усі розділи роботи послідовні та логічні, що дозволяє чітко розуміти викладений матеріал в рамках тематики кваліфікаційної роботи. Графічний матеріал дозволяє наочно побачити доцільність та ефективність рішень, які були прийняті за основу для досягнення поставленої мети.

8. Інші зауваження Окремі описи в пояснювальній записці подано занадто деталізовано, що ускладнює сприйняття матеріалу фахівцями в обраній предметній галузі.

9. Оцінка кваліфікаційної роботи Враховуючи всі позитивні та негативні сторони представленої кваліфікаційної роботи, можна зробити висновок, що вона заслуговує оцінку «відмінно».

РЕЦЕНЗЕНТ (прізвище, ім'я, по батькові, посада, місце роботи)
Лисенко Сергій Миколайович д. т. н. доцент КІСП

« 07 » 06 2021. _____ (підпис)

UNICHECK

User name:
**Кафедра кибербезпеки**

Check ID:
**1008169768**

Check date:
**04.06.2021 09:32:27 EEST**

Check type:
**Doc vs Internet**

Report date:
**04.06.2021 09:33:40 EEST**

User ID:
**100005590**

File name: **Кваліфікаційна робота Стопчак**

Page count: **55** Word count: **9013** Character count: **74365** File size: **1.39 MB** File ID: **1008248562**

# 0.9% Matches

Highest match: **0.3%** with Internet source (**https://ela.kpi.ua/bitstream/123456789/28660/1/Ivanenko_bakalavr.pdf**)

**0.9% Internet sources** **5** ..... Page 57

**No Library search was conducted**

# 0% Quotes

**Exclusion of quotes is off**

**Exclusion of references is off**

# 0% Exclusions

**No exclusions**

# Anti-Plagiarism v-15.257

**Максимальное совпадение с одним документом 0.0%**

Словари проверки: en_US, ru_RU, ua_UA. **Ошибок в документах: 10%**

| | Документ | | Суммарное совпадение по Базе Данных | |
|---|---|---|---|---|
| ID: 92260<br>Название: Система захисту від витоку даних Відділу протидії кіберзлочинам Департаменту кіберполіції Національної поліції України<br>Добавлено в БД: 2021-06-04<br>Авторы: О.О. Стопчак<br>Руководители: І.В. Муляр<br>Консультанты:<br>Опоненты: | Символы | Лексемы | Символы | Лексемы |
| | 63288 | 491 | 240 (0%) | 5 (1%) |

Источник плагиата

| ID | Описание | Наличие плагиата в документе | |
|---|---|---|---|
| | | Символы | Лексемы |