ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ ТА ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«СИСТЕМА МОНІТОРИНГУ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ ПРИ ВИКОРИСТАННІ КОРПОРАТИВНИХ ЧАТІВ У TELEGRAM»

на здобуття освітнього ступеня магістр
зі спеціальності 124 Системний аналіз
(код,найменування спеціальності)
освітньо-професійної програми Інтелектуальні системи управління
(назва)

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Богдан ЧЕРЕВАТИЙ
(підпис) *(ім'я, ПРІЗВИЩЕ здобувача)*

Виконав:
здобувач вищої освіти
група САДМ-61 Богдан ЧЕРЕВАТИЙ
(ім'я, ПРІЗВИЩЕ)
Керівник д.т.н., проф. Олесій ШУШУРА
(ім'я, ПРІЗВИЩЕ)

Рецензент: к.т.н., доцент Наталія ЛАЩЕВСЬКА
(ім'я, ПРІЗВИЩЕ)

Київ 2026

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут Інформаційних технологій

Кафедра Інформаційних систем та технологій
Ступінь вищої освіти магістр
Спеціальність 124 Системний аналіз
Освітньо-професійна прорама Інтелектуальні системи управління

ЗАТВЕРДЖУЮ

Завідувач кафедрою ІСТ
Каміла СТОРЧАК

“____” _____2025 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Череватому Богдану Сергійовичу
(прізвище, ім’я, по батькові здобувача)

1.Тема кваліфікаційної роботи: Система моніторингу інформаційної безпеки при використанні корпоративних чатів у Telegram
керівник кваліфікаційної роботи: Олексій ШУШУРА д.т.н., проф.
(ім’я, ПРІЗВИЩЕ, науковий ступінь, вчене звання)
затверджені наказом Державного університету інформаційно-комунікаційних технологій від “30” жовтня 2025 р. № 467

2. Строк подання кваліфікаційної роботи «26» грудня 2025 р.

3. Вихідні дані кваліфікаційної роботи:
1. Аналіз тематичної галузі та вимог до системи моніторингу інформаційної безпеки.
2. Теоретичні основи та проектування системи моніторингу інформаційної безпеки.
3. Проектування та реалізація системи моніторингу інформаційної безпеки корпоративних чатів у Telegram.
4. Науково-технічна література.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити):
1. Аналіз інформаційної безпеки корпоративних чатів у Telegram.
2. Теоретичні основи та проектування системи моніторингу.

3. Розроблення та оцінка ефективності прототипу системи моніторингу.

5.Перелік ілюстраційного матеріалу: *презентація*

6. Дата видачі завдання «30» жовтня 2025р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Підбір технічної літератури	01.11.2025	
2.	Проектування вимог до системи моніторингу інформаційної безпеки	07.11.2025	
3.	Система моніторингу інформаційної безпеки	10.11.2025	
4.	Розроблення прототипу системи моніторингу	22.11.2025	
5.	Висновки по роботі	23.11.2025	
6.	Розробка демонстраційних матеріалів, доповідь.	26.11.2025	
7.	Оформлення магістерської роботи	04.12.2025	

Здобувач вищої освіти _____ Богдан ЧЕРЕВАТИЙ
(підпис) *(ім'я, ПРІЗВИЩЕ)*

Керівник кваліфікаційної роботи _____ Олексій ШУШУРА
(підпис) *(ім'я, ПРІЗВИЩЕ)*

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня магістра: 65 стор., 9 рис., 30 джерел.

Мета роботи – розробити систему моніторингу, призначену для виявлення та запобігання порушенням інформаційної безпеки в корпоративних чатах месенджера Telegram.

Об'єкт дослідження – процес контролю інформаційної безпеки корпоративного чату в месенджері Telegram.

Предмет дослідження – моделі, методи, алгоритми та програмне забезпечення системи моніторингу інформаційної безпеки корпоративних чатів у месенджері Telegram.

Короткий зміст роботи. У першому розділі магістерської роботи розглянуто особливості корпоративної комунікації у месенджері Telegram, проаналізовано його архітектуру, механізми безпеки та класифіковано основні загрози інформаційній безпеці при використанні корпоративних чатів. Оцінено наявні підходи та інструменти моніторингу безпеки в месенджерах і сформульовано вимоги до системи моніторингу Telegram-чатів.

У другому розділі наведено теоретичні основи побудови системи моніторингу інформаційної безпеки корпоративних чатів, розроблено концептуальну модель та архітектуру програмної системи, спроектовано структуру бази даних та інформаційні потоки, а також описано алгоритми аналізу подій і виявлення порушень політик безпеки.

У третьому розділі подано опис реалізації прототипу системи на основі Telegram-боту, структуру основних програмних модулів та логіку їх взаємодії. Наведено приклади роботи системи, результати експериментальної перевірки та виконано аналіз ефективності запропонованого підходу до моніторингу інформаційної безпеки в корпоративних чатах Telegram.

КЛЮЧОВІ СЛОВА: КОРПОРАТИВНІ ЧАТИ, ПОЛІТИКА БЕЗПЕКИ, МОНІТОРИНГ, ІНФОРМАЦІЙНА БЕЗПЕКА, ІНЦЕДЕНТИ, БОТ, TELEGRAM

ABSTRACT

Text part of the qualification work for obtaining a master's degree: 65 pages, 9 figures, 30 sources.

The purpose of the work is to develop a monitoring system designed to detect and prevent information security breaches in corporate Telegram messenger chats.

The object of research is the process of controlling information security in corporate chat in the Telegram messenger.

The subject of research is the models, methods, algorithms, and software for monitoring information security in corporate chats in the Telegram messenger.

Brief summary of the work. The first chapter of the master's thesis examines the features of corporate communication in the Telegram messenger, analyzes its architecture and security mechanisms, and classifies the main threats to information security when using corporate chats. Existing approaches and tools for monitoring security in messengers are evaluated, and requirements for a Telegram chat monitoring system are formulated.

The second chapter presents the theoretical foundations for building a corporate chat information security monitoring system, develops a conceptual model and architecture for the software system, designs the database structure and information flows, and describes algorithms for analyzing events and detecting security policy violations.

The third chapter describes the implementation of a prototype system based on a Telegram bot, the structure of the main software modules, and the logic of their interaction. Examples of the system's operation and the results of experimental testing are provided, and an analysis of the effectiveness of the proposed approach to monitoring information security in corporate Telegram chats is performed.

KEYWORDS: CORPORATE CHATS, SECURITY POLICY, MONITORING, INFORMATION SECURITY, INCIDENTS, BOT, TELEGRAM.

ЗМІСТ

ВСТУП

Актуальність теми. У сучасному цифровому середовищі корпоративна комунікація все частіше здійснюється за допомогою месенджерів, які забезпечують швидкість обміну інформацією, мобільність та зручність взаємодії співробітників. Telegram став одним з найпопулярніших інструментів у бізнессередовищі завдяки своїй функціональності, кросплатформенній сумісності та розширеним можливостям інтеграції. Однак активне використання месенджерів для передачі офіційних даних супроводжується ризиками порушення інформаційної безпеки, витоку конфіденційної інформації, соціальної інженерії та несанкціонованого доступу.

Відсутність централізованих механізмів контролю політики безпеки в Telegram вимагає створення спеціалізованих систем моніторингу, які забезпечують виявлення інцидентів у режимі реального часу та підвищують безпеку корпоративних чатів. Таким чином, розробка системи моніторингу інформаційної безпеки корпоративних чатів в Telegram є актуальним науково-практичним завданням.

Мета роботи – розробити систему моніторингу, призначену для виявлення та запобігання порушенням інформаційної безпеки в корпоративних чатах месенджера Telegram.

Для досягнення цієї мети необхідно вирішити такі завдання:

– проаналізувати особливості корпоративної комунікації та механізми Telegram у контексті інформаційної безпеки;

– виявити загрози існуючих підходів до захисту корпоративних чатів;

– сформулювати вимоги до системи моніторингу інформаційної архітектури чатів Telegram;

– розробити концептуальну модель та архітектуру системи;

– реалізувати програмний прототип системи моніторингу;

– провести експериментальну оцінку ефективності запропонованої системи та проаналізувати її переваги над існуючими рішеннями;

Об'єкт дослідження – процес забезпечення інформаційної безпеки в корпоративних комунікаціях.

Предмет дослідження – методи, моделі та програмні засоби моніторингу інформаційної безпеки в корпоративних чатах Telegram.

Методи дослідження. У цій роботі були використані такі методи дослідження:

– системний аналіз для дослідження предметної області;

– методи оцінки ризиків інформаційної безпеки;

– моделювання системної архітектури;

– методи аналізу текстових даних та подій у чатах;

– експериментальні методи тестування розробленого прототипу;

Наукова новизна одержаних результатів полягає у розробці спеціалізованої моделі та архітектури системи моніторингу інформаційної безпеки корпоративних чатів у месенджері Telegram, орієнтованої на використання месенджера як корпоративного каналу зв'язку. Запропонований підхід поєднує сигнатурні методи контролю з поведінковим та контекстно-семантичним аналізом подій, а також формалізує вимоги, структуру даних і логіку взаємодії компонентів системи для автоматичного виявлення порушень політик безпеки та подальшої інтеграції рішення в існуючу інфраструктуру підприємства.

Практична значущість одержаних результатів полягає в можливості використання розробленої системи для підвищення рівня безпеки корпоративних чатів, запобігання витоку інформації та поліпшення контролю за дотриманням політик безпеки в організації. Система може бути адаптована до різних бізнес-середовищ і масштабована відповідно до потреб підприємства.

Апробація результатів магістерської роботи. Основні положення і результати магістерської роботи доповідались на науково практичних конференціях, що проходили на базі Державного університету інформаційно-комунікаційних технологій та Державного університету «Житомирська політехніка».

1 АНАЛІЗ ТЕМАТИЧНОЇ ГАЛУЗІ ТА ВИМОГ ДО СИСТЕМИ МОНІТОРИНГУ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ

1.1 Особливості корпоративних комунікацій в сучасних організаціях

Розвиток інформаційних технологій призвів до глибоких змін у способах взаємодії людей як у приватному, так і в корпоративному середовищі. Традиційні засоби комунікації, такі як електронна пошта та телефонні дзвінки, поступово замінюються більш гнучкими, динамічними та технологічно досконалими засобами. Одним з таких засобів є інтернет-месенджери, які завдяки своїй мобільності, швидкості та широкому функціоналу відіграють ключову роль у підтримці ефективності внутрішніх бізнес-процесів.

Корпоративна комунікація передбачає обмін повідомленнями між співробітниками, керівництвом, відділами та зовнішніми партнерами. Вона охоплює як оперативні завдання (наприклад, координацію робочих процесів), так і стратегічні аспекти діяльності компанії. Ефективність таких процесів значною мірою залежить від швидкості доступу до інформації та зручності її передачі, що визначає вибір засобів комунікації.

Використання месенджерів у корпоративному середовищі дозволяє забезпечити швидкий обмін робочою інформацією в режимі реального часу, організовувати групи за відділами, проєктами чи функціональними напрямами, створювати централізовані канали інформування, інтегрувати автоматизовані інструменти на кшталт чат-ботів, передавати документи, мультимедійні матеріали та робочі файли, а також ефективно координувати діяльність віддалених співробітників.

Переваги мобільності та доступності сприяють тому, що месенджери стають основним засобом комунікації в організаціях різного розміру – , від малих підприємств до міжнародних корпорацій. Однак така популярність супроводжується підвищеними ризиками.

На відміну від спеціалізованих корпоративних платформ, месенджери часто не мають вбудованих механізмів дотримання стандартів інформаційної безпеки, що ставить додаткові вимоги до їх використання та контролю. Зокрема, за відсутності чітко визначених політик або функціоналу моніторингу існує ризик порушення конфіденційності, несвоєчасного виявлення інцидентів безпеки та витоку критично важливої інформації.

Таким чином, особливості корпоративної комунікації в сучасних умовах створюють необхідність додаткових заходів контролю, які забезпечать дотримання політик безпеки та захист офіційної інформації незалежно від обраного інструменту.

1.2 Telegram як платформа для корпоративних чатів: архітектура, API, механізми безпеки

Telegram є одним з найпопулярніших месенджерів, що використовуються в корпоративному середовищі, завдяки поєднанню швидкості, розширених функцій та кросплатформної підтримки. Його архітектура розроблена для забезпечення стійкості до зовнішніх атак, масштабованості та доступності сервісу навіть під великим навантаженням.

Telegram використовує власний криптографічний протокол MTProto, який поєднує симетричне та асиметричне шифрування і мінімізує ризик несанкціонованого доступу до даних. Повідомлення передаються через розподілені центри обробки даних, що підвищує загальну стабільність системи.

Це дозволяє користувачам отримувати доступ до своїх чатів з будь-якого пристрою, незалежно від того, де було створено повідомлення. Хмарні чати унікальні тим, що їхній вміст синхронізується на всіх пристроях, пов'язаних з обліковим записом.

Telegram використовує модель шифрування клієнт-сервер, яка забезпечує базову конфіденційність. Однак групові чати не підтримують шифрування «від кінця до кінця», що створює потенційні можливості для доступу до даних на сервері.

Завдяки Telegram Bot API розробники можуть створювати спеціалізовані інструменти для автоматизації бізнес-процесів, моніторингу подій та аналізу даних у чатах. Це дозволяє розширити базову функціональність і впровадити інструменти для конкретних потреб вашої компанії.

Telegram дозволяє створювати супергрупи з до 200 000 учасників, що робить його придатним для великих корпоративних структур з великою кількістю співробітників.

Незважаючи на високий рівень технологічної досконалості, Telegram має ряд обмежень в контексті корпоративної безпеки. Зокрема, платформа не надає системи аудиту, моніторингу активності користувачів, централізованого логування або інтегрованої підтримки корпоративних політик безпеки. Ці фактори роблять створення спеціалізованих систем моніторингу нагальним питанням.

1.3 Загрози інформаційній безпеці при використанні месенджерів

Використання месенджерів у корпоративному середовищі супроводжується низкою ризиків, які можна класифікувати за технічними, організаційними та людськими факторами.

Технічні ризики пов'язані з особливостями месенджерів та пристроїв користувачів.

До технічних загроз належать ризик перехоплення даних через ненадійні мережеві з'єднання, використання шифрування типу "клієнт-сервер", яке не забезпечує наскрізного шифрування повідомлень у групових чатах, наявність вразливостей у клієнтських додатках, що можуть бути використані для несанкціонованого доступу, можливість компрометації токенів ботів, особливо за

умови їх розміщення в ненадійному середовищі, а також небезпека використання модифікованих клієнтів Telegram, які можуть містити шкідливі компоненти.

Організаційні загрози, що є наслідком внутрішніх процесів у компанії, проявляються у відсутності централізованих правил використання месенджерів, неконтрольованому доступі співробітників до робочих чатів, браку інструментів аудиту, практиці пересилання конфіденційних документів в особисті чати та недотриманні встановлених правил класифікації даних.

Людський фактор, з яким пов'язана більшість інцидентів інформаційної безпеки, включає помилки під час надсилання повідомлень, передавання службової інформації третім особам, успішні фішинг-атаки, спрямовані на співробітників, а також випадкове розголошення персональних або фінансових даних.

Сукупність цих ризиків підкреслює необхідність створення автоматизованої системи моніторингу, здатної виявляти потенційно небезпечні дії в корпоративних чатах.

1.4 Політика інформаційної безпеки в корпоративних чатах: принципи, обмеження та відповідальність користувачів

Політики інформаційної безпеки є невід'ємною складовою ефективного функціонування корпоративних комунікаційних систем. При використанні месенджерів, таких як Telegram, вони відіграють ключову роль у регулюванні поведінки співробітників, визначенні правил доступу до конфіденційних даних та запобіганні інцидентам, пов'язаним з витоком інформації. Будь-яка організація, яка прагне забезпечити належний рівень захисту своїх інформаційних активів, повинна впровадити чітко визначені та формалізовані правила використання корпоративних чатів.

Формування політик інформаційної безпеки базується на міжнародних стандартах, таких як ISO/IEC 27001, ISO/IEC 27002 та рекомендаціях NIST.

У контексті корпоративного використання Telegram ці принципи відіграють фундаментальну роль, оскільки сам месенджер не містить механізмів повного контролю за діями користувачів, а отже, організація повинна компенсувати цей недолік власними заходами.

Особлива увага приділяється регулюванню ділового листування. Нечіткі формулювання, неправильна передача інформації або використання нестандартного словника можуть стати джерелом конфліктів, непорозумінь або навіть юридичних ризиків. Тому правила спілкування не менш важливі, ніж технічні вимоги.

Користувачі корпоративних чатів несуть відповідальність за:

- дотримання встановлених політик;
- правильне використання офіційної інформації;
- запобігання розголошенню конфіденційних даних;
- негайне повідомлення про будь-які підозрілі дії або інциденти.

Відсутність відповідальності з боку співробітників або недостатня обізнаність можуть перетворити навіть найбезпечніший месенджер на джерело значних загроз. Саме тому більшість корпоративних політик супроводжуються обов'язковим навчанням співробітників та регулярним моніторингом їхньої діяльності.

1.5 Аналіз сучасних підходів та систем моніторингу безпеки в месенджерах

Оскільки корпоративні месенджери не завжди мають вбудовані функції безпеки, на ринку з'явився широкий спектр інструментів, що частково компенсують ці недоліки. У цьому розділі ми розглянемо основні підходи, що використовуються для моніторингу безпеки в системах чату, а також їхні обмеження.

Підхід 1. Вбудовані інструменти модерації.

Боти-модератори Telegram надають базові можливості, серед яких:

- видалення повідомлень за певними правилами;
- фільтрування спаму;
- обмеження доступу для нових учасників;
- автоматична модерація заборонених типів контенту.

Однак ці інструменти не призначені для забезпечення інформаційної безпеки, оскільки:

- вони не аналізують корпоративні політики;
- вони не виявляють внутрішніх витоків;
- вони не забезпечують аудит дій користувачів;
- вони не ведуть журнал подій у зручному форматі.

Іншими словами, вони функціонують більше як системи модерації, ніж як системи безпеки.

Підхід 2. Системи DLP.

Системи запобігання втраті даних призначені для контролю обміну інформацією всередині компанії. Вони виконують:

- аналіз тексту;
- виявлення ключових слів;
- контроль передачі файлів;
- ідентифікацію конфіденційних даних (номери карток, повні імена, ідентифікаційні номери платників податків тощо).

Однак стандартні системи DLP не мають прямої інтеграції з API Telegram, а тому:

- не можуть аналізувати групові чати в режимі реального часу;
- не мають доступу до історії повідомлень у хмарі;
- не працюють повноцінно в мобільному сегменті.

Їх інтеграція можлива лише за допомогою складних проміжних рішень, що значно збільшує вартість впровадження.

Підхід 3. Системи SIEM/SOAR.

Системи управління інформацією та подіями безпеки (SIEM) та автоматизованого реагування (SOAR) стали стандартом кібербезпеки для великих компаній.

Однак використання SIEM у Telegram обмежене:

- SIEM може обробляти тільки події, що надходять через API;
- Telegram не передає детальну телеметрію про поведінку користувачів;
- інциденти в чатах можуть бути пропущені через затримки в зборі інформації.

Таким чином, системи SIEM ефективні лише як додатковий рівень, але не як основний інструмент для моніторингу корпоративних чатів.

Підхід 4. Спеціалізовані корпоративні рішення.

Великі компанії іноді створюють власні інструменти, які:

- підключаються до Telegram через ботів;
- аналізують повідомлення;
- виявляють підозрілу активність;
- збирають статистику.

Недоліками таких рішень є:

- високі витрати на розробку;
- складність обслуговування;
- необхідність залучення фахівців з кібербезпеки та машинного навчання;
- низька гнучкість у масштабуванні.

Тому на ринку все ще бракує комплексних інструментів, які б повністю задовольняли потреби моніторингу корпоративних чатів Telegram.

1.6 Визначення основних вимог до системи моніторингу чатів Telegram

З урахуванням аналізу особливостей корпоративного спілкування в Telegram, загроз інформаційній безпеці та існуючих підходів до моніторингу формується набір вимог до системи, яка повинна забезпечувати повний контроль за робочим чатом без порушення звичного режиму роботи користувачів. Перш за все, система повинна підтримувати моніторинг в режимі реального часу, тобто здатність швидко обробляти події, що надходять з корпоративного чату, і миттєво реагувати на потенційні інциденти. Затримки у виявленні порушень можуть призвести до витоку конфіденційної інформації або ескалації внутрішніх загроз, тому вимога щодо швидкості є однією з основних.

Важливою є також здатність системи проводити глибокий аналіз текстових повідомлень. Це передбачає не тільки пошук конкретних ключових фраз, але й виявлення посилань на особисті дані, технічну або фінансову інформацію, яка, згідно з корпоративною політикою, не повинна передаватися у відкритій формі. У поєднанні з цим повинен бути механізм перевірки файлів, що передаються в чаті: система повинна виявляти документи, які потенційно містять конфіденційну інформацію, і реагувати відповідно до налаштованих політик – від реєстрації інцидентів до блокування доступу.

Окремий набір вимог стосується аудиту дій користувачів. Система повинна реєструвати додавання та видалення учасників, зміни їхніх прав доступу, надання статусу адміністратора та інші критичні дії, що впливають на рівень довіри в комунікаційному середовищі. На основі цих даних можна створювати поведінкові профілі та виявляти аномалії в діяльності окремих користувачів. Для ефективного управління всіма цими процесами необхідний централізований контроль над політиками безпеки: адміністратор повинен мати можливість змінювати правила моніторингу, встановлювати порогові значення та диференціювати вимоги для різних груп користувачів або типів чатів.

Не менш важливою є наявність механізму оперативного повідомлення про інциденти. У разі виявлення порушення система повинна інформувати

відповідальних осіб через зручні канали (повідомлення в окремому чаті адміністратора, електронна пошта, інтеграція з іншими системами безпеки), надаючи їм достатню контекстну інформацію для прийняття рішень. Водночас необхідно вести повний журнал подій: реєстрація повідомлень, інцидентів, автоматичних відповідей та дій адміністратора створює основу для подальшого аудиту та ретроспективного аналізу.

Нарешті, важливо, щоб система була масштабованою і могла інтегруватися з іншими корпоративними рішеннями. Це означає здатність обробляти зростаючі потоки повідомлень без значної втрати продуктивності, підтримувати кілька чат-з'єднань або навіть цілу інфраструктуру каналів, а також обмінюватися даними із зовнішніми системами класу SIEM, каталогами користувачів або внутрішніми інформаційними службами. Ці вимоги формують технічну основу майбутнього програмного комплексу і визначають перелік функціональних можливостей, які повинні бути реалізовані в розробленій системі моніторингу інформаційної безпеки.

1.7 Постановка задачі

На основі проведеного аналізу особливостей використання месенджера Telegram у корпоративних цілях встановлено, що, з одного боку, він є зручним та ефективним інструментом комунікації завдяки мобільності, швидкості передачі даних, хмарній архітектурі та підтримці розширень через Bot API. З іншого боку, така популярність та гнучкість породжують додаткові ризики для інформаційної безпеки, оскільки Telegram, як і більшість загальнодоступних месенджерів, спочатку не проектувався як повноцінна корпоративна платформа із вбудованими механізмами аудиту, централізованого журналювання та контролю відповідності політикам безпеки.

Аналіз загроз показав, що технічні, організаційні та пов'язані з людським фактором ризики у сукупності створюють істотний потенціал для витоку службової інформації через корпоративні чати. Виявлено, що існуючі рішення – від простих модераційних ботів до DLP- та SIEM/SOAR-систем – не забезпечують комплексного, гнучкого й економічно доцільного інструменту саме для моніторингу корпоративних чатів у Telegram. Це зумовлює наявність прогалини між реальними потребами організацій у контролі комунікацій та можливостями наявних засобів захисту, що й формує науково-прикладну проблему даної роботи.

З огляду на це постановка задачі полягає у розробленні спеціалізованої системи моніторингу інформаційної безпеки для корпоративних чатів у Telegram, яка б забезпечувала моніторинг у режимі реального часу, глибокий аналіз тексту та вкладень, аудит дій користувачів, централізоване керування політиками та оперативне сповіщення про інциденти, а також могла інтегруватися з існуючою корпоративною інфраструктурою. Така система має працювати поверх наявного функціоналу Telegram, не змінюючи звичний процес комунікації для користувачів, але додаючи до нього рівень прозорості та контрольованості з боку служби інформаційної безпеки.

Відповідно до сформульованої проблеми метою роботи є розроблення системи моніторингу інформаційної безпеки корпоративних чатів у Telegram, здатної оперативно виявляти порушення політик та інциденти безпеки й забезпечувати підтримку ухвалення рішень адміністраторами без додаткового навантаження на користувачів.

Для досягнення поставленої мети необхідно розв'язати такі основні завдання:

1. Виконати аналіз особливостей корпоративної комунікації в Telegram та класифікувати основні загрози інформаційній безпеці при використанні корпоративних чатів.
2. Сформулювати вимоги до системи моніторингу, які б відображали потреби реального корпоративного середовища.
3. Розробити концептуальну модель і архітектуру програмної системи моніторингу Telegram-чатів, включно зі структурою бази даних та інформаційних потоків.

4. Спроектувати та реалізувати прототип системи на базі Telegram-боту з підтримкою аналізу тексту, вкладень і поведінкових характеристик користувачів.
5. Провести експериментальну перевірку роботи прототипу, оцінити ефективність виявлення порушень політик та визначити перспективи подальшого розвитку системи.

Висновки до розділу 1

У першому розділі виконано комплексний аналіз використання месенджера Telegram у корпоративних цілях та обґрунтовано необхідність створення спеціалізованої системи моніторингу інформаційної безпеки. Показано, що месенджери стали ключовим інструментом корпоративної комунікації завдяки мобільності, швидкості та зручності, але водночас створюють нові канали для витоку даних. Telegram розглянуто як платформу для корпоративних чатів: відзначено його переваги (хмарна архітектура, Bot API, підтримка великих груп) і обмеження з погляду ІБ (відсутність повного end-to-end шифрування для груп, вбудованого аудиту, централізованого журналювання та підтримки корпоративних політик).

Проаналізовано загрози інформаційній безпеці при використанні месенджерів, які згруповано на технічні, організаційні та пов'язані з людським фактором. Наголошено, що без формалізованих політик інформаційної безпеки та відповідальності користувачів навіть технологічно розвинуті платформи не гарантують належного захисту. Окремо розглянуто наявні підходи до моніторингу (модераційні боти, DLP, SIEM/SOAR, внутрішні корпоративні рішення) і зроблено висновок, що жоден із них не забезпечує комплексного, гнучкого й доступного контролю саме для корпоративних Telegram-чатів.

На основі проведеного аналізу сформульовано вимоги до системи моніторингу: підтримка моніторингу в реальному часі, глибокий аналіз тексту та

вкладень, аудит активності користувачів, централізоване керування політиками, механізми сповіщення про інциденти, ведення журналів подій, масштабованість та інтеграція з корпоративною інфраструктурою. Сформульовано мету роботи – розробити систему моніторингу інформаційної безпеки корпоративних чатів у Telegram, здатну оперативно виявляти порушення політик та визначено основні завдання дослідження. Отримані результати закладають теоретичне й практичне підґрунтя для подальшого проектування та реалізації системи в наступних розділах.

2. ТЕОРЕТИЧНІ ОСНОВИ ТА ПРОЕКТУВАННЯ СИСТЕМИ МОНІТОРИНГУ ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1 Теоретичні основи моніторингу інформаційної безпеки

Моніторинг інформаційної безпеки – це сукупність організаційних, технічних та аналітичних заходів, спрямованих на своєчасне виявлення, фіксацію та реагування на події та інциденти, що загрожують конфіденційності, цілісності або доступності інформаційних активів. У теоретичному плані моніторинг розглядається як безперервний цикл, що включає збір даних, кореляцію, виявлення аномалій або порушень політики, оцінку ризиків та ініціювання відповідних контрзаходів.

Подія – окрема дія або повідомлення (наприклад, надсилання повідомлення в чаті, передача файлу, зміна ролі користувача), що реєструється системою.

Інцидент – подія або сукупність подій, які вважаються порушенням політики безпеки або можуть призвести до негативних наслідків.

Сповіщення – повідомлення оператору/адміністратору про потенційний інцидент.

Кореляція подій – процес об'єднання декількох подій для виявлення більш складних атак або моделей поведінки.

Три етапи моніторингу: (1) збір і нормалізація журналів, (2) виявлення (правила/аналітика/машинне навчання), (3) реагування та документація.

У науковій літературі розрізняють дві основні парадигми виявлення інцидентів: на основі правил (на основі сигнатур) та поведінкова/аналітична (на основі аномалій, поведінкове виявлення). Перший підхід добре працює для відомих моделей (наприклад, прості правила за ключовими словами), тоді як другий здатний виявляти нові, невідомі атаки шляхом ідентифікації відхилень від нормальної поведінки.

Підходи до моніторингу в корпоративних месенджерах повинні враховувати особливості комунікацій (короткі повідомлення, лаконічні формати, велика

кількість мультимедійних вкладень), а також вимоги до конфіденційності (обмеження на збір персональних даних та їх анонімізацію).

2.2 Моделі виявлення інцидентів та оцінки ризиків у корпоративних Telegram-чатах

Виявлення сигнатур базується на наборі заздалегідь визначених правил або шаблонів. У контексті чатів Telegram це можуть бути:

- правила для ключових слів/фраз (наприклад, «пароль», «номер картки», «конфіденційно»);
- регулярні вирази для виявлення номерів карток, ІПН, адрес електронної пошти;
- правила для типів вкладень (виключення .exe, архівів, захищених паролем, тощо);
- правила для частоти дій (занадто часті передачі файлів, масові передачі до зовнішніх чатів).

Переваги: простота реалізації, висока інтерпретованість. Недоліки: необхідність оновлення правил, низька адаптованість до нових загроз, велика кількість помилкових спрацьовувань/помилкових відмов при неправильній конфігурації.

Поведінкове виявлення передбачає побудову моделей нормальної активності та виявлення відхилень. Методи:

Статистичний (пороговий): на основі частоти повідомлень, часових моделей, розмірів вкладень.

Неконтрольоване ML: кластеризація (k-means, DBSCAN), методи зменшення розмірності (PCA), алгоритми виявлення аномалій (Isolation Forest, One-Class SVM).

Наглядовий ML: якщо доступна історія інцидентів з мітками, класифікатори (логістична регресія, випадковий ліс, градієнтне підсилення, нейронні мережі) можуть бути навчені прогнозувати ймовірність інциденту.

Поведінкові моделі дозволяють ідентифікувати нетипові сценарії (наприклад, користувач, який рідко надсилає файли, раптово надсилає великий архів; аномалії в часі активності; одночасний вхід з декількох IP-адрес).

Оптимальна архітектура зазвичай поєднує підходи на основі сигнатур та поведінкові підходи: правила швидко відфільтровують очевидні порушення, а поведінкова аналітика обробляє більш складні шаблони. Крім того, кореляція із зовнішніми джерелами (чорні списки, сервіси репутації) покращує якість виявлення.

Кожен потенційний інцидент оцінюється за допомогою моделі ризику:

Ризик = Імовірність × Наслідок.

Імовірність розраховується на основі вхідних сигналів (наприклад, модель ML видає ймовірність інциденту), а наслідки розраховуються на основі категорії даних, масштабу поширення та ролей залучених користувачів. Результатом є числовий бал (0-100), який визначає рівень реагування (інформаційне повідомлення, блокування дії, дзвінок для розслідування).

2.3 Концептуальна модель системи моніторингу чатів Telegram

Концептуальна модель описує основні компоненти системи та їх взаємодію на логічному рівні. Запропонована модель містить такі блоки:

- джерела даних: повідомлення користувачів, метадані (час, автор, чат), вкладення (файли), адміністративні події (зміни ролей);
- модуль збору: підключення через Telegram Bot API або інші шляхи інтеграції, нормалізація даних, тимчасове кешування;
- система черги/потокової передачі: проміжний рівень, що забезпечує надійну доставку та буферизацію подій;

- модуль попередньої обробки: очищення тексту, токенізація, видалення стоп-слів, визначення мови, розпізнавання типу файлу, вилучення метаданих;
- модуль правил: застосування правил і політик підписів;
- аналітичний модуль: поведінкова аналітика, ML-моделі, кореляція подій, формування ризиків;
- модуль реагування: повідомлення адміністратора, блокування операцій, автоматичні відповіді/попередження користувачеві;
- система зберігання: база даних для хронологічної реєстрації подій, зберігання файлів;
- інтерфейс адміністратора: панель для перегляду інцидентів, редагування політик, формування звітів;
- інтеграційний рівень: підключення до SIEM/SOC, LDAP/AD, систем розслідування.

Ця модель дозволяє розподілити обов'язки між компонентами та забезпечує гнучкість масштабування.

Діаграма послідовності дій моделі Telegram-боту показана на рис. 2.1.

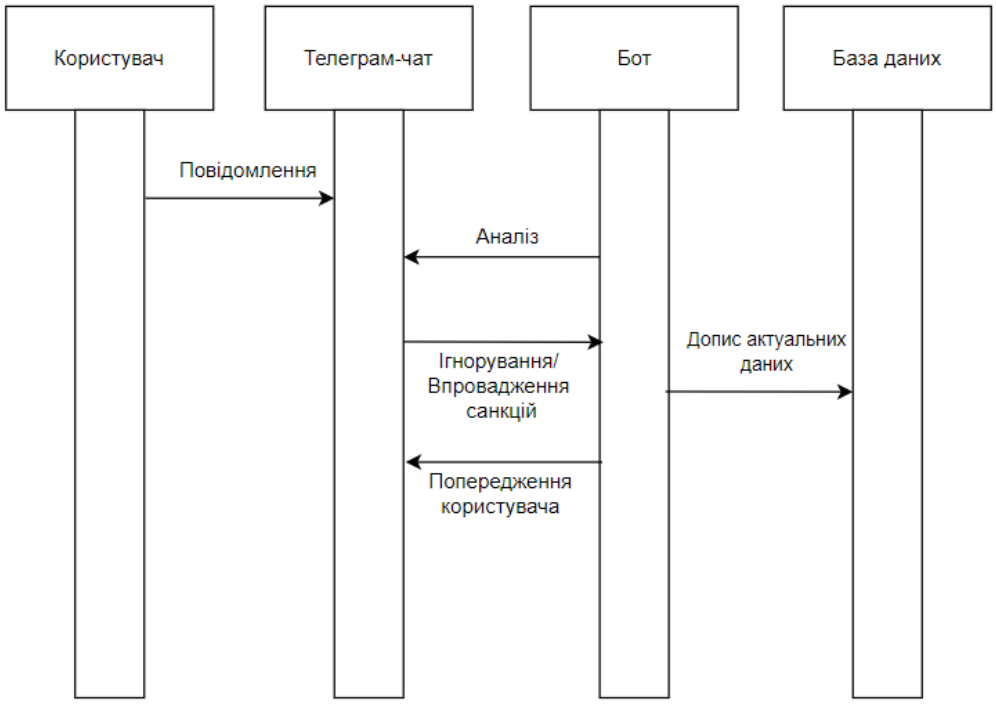


Рис. 2.1 Послідовність дій моделі боту Telegram

2.4 Архітектура програмної системи моніторингу

Архітектура пропонованої програмної системи моніторингу інформаційної безпеки корпоративних чатів у Telegram базується на принципах модульності, мікросервісів та горизонтального масштабування. Це означає, що вся система поділена на окремі логічні компоненти, кожен з яких виконує свою чітко визначену функцію, а взаємодія між ними здійснюється через стандартизовані інтерфейси. Такий підхід забезпечує гнучкість, спрощує оновлення окремих частин системи та дозволяє адаптувати рішення до різних навантажень і сценаріїв роботи.

Початковим ланцюжком в архітектурі є компонент інтеграції Telegram, який виступає в ролі адаптера або збирача подій. Він підключається до платформи через Telegram Bot API, отримує всі події з корпоративного чату – текстові повідомлення, файли, сервісні повідомлення, дії адміністратора – і перетворює їх в уніфікований внутрішній формат (зазвичай об'єкти JSON з чітко визначеними полями). Це дозволяє відокремити особливості протоколу Telegram від подальших етапів обробки і забезпечує єдину презентацію даних для всіх інших модулів.

Щоб система могла стабільно працювати під час пікових навантажень, між модулем збору подій і аналітичними службами розміщується проміжний шар у вигляді брокера повідомлень. Такий компонент реалізується, наприклад, на основі Kafka або RabbitMQ і відповідає за асинхронну обробку потоку подій, гарантовану доставку повідомлень і буферизацію в моменти, коли швидкість надходження даних перевищує можливості для негайної обробки. Завдяки цьому модуль аналізу не перевантажується, а система залишається працездатною навіть у разі короткочасних збоїв окремих служб.

Окремим шаром архітектури є служба попередньої обробки. Саме тут виконуються операції, пов'язані з підготовкою складних типів даних до аналізу: розпакування та перекодування вкладень, розпізнавання тексту в зображеннях, вилучення вмісту з PDF та офісних документів, нормалізація структур подій та збагачення додатковими атрибутами – інформацією про пристрій, IP-адресу,

геолокацію тощо. Таким чином, наступні рівні отримують уніфіковані та максимально інформативні події, які можна обробляти в єдиному форматі.

Центральним елементом логіки системи є механізм правил. Цей компонент містить політики безпеки, які визначають, як слід інтерпретувати певні події. Модуль дозволяє динамічно додавати та змінювати правила без перезапуску всієї системи і підтримує використання складних логічних виразів, часових вікон та комбінованих умов. Завдяки цьому може відфільтровувати очевидні порушення навіть на початковому етапі фільтрації, не перевантажуючи більш ресурсоємні аналітичні модулі.

Наступний рівень складається з сервісів аналітики та машинного навчання. Вони відповідають за виявлення аномалій у поведінці користувачів, класифікацію повідомлень за рівнем ризику та розпізнавання прихованих закономірностей у комунікації. Для підтримки таких моделей використовується сховище ознак, яке дозволяє накопичувати та повторно використовувати обчислені характеристики повідомлень, користувачів та їхньої активності. Для навчання та валідації моделей можуть бути виділені окремі модулі, що дає можливість поступово покращувати якість виявлення інцидентів, не заважаючи роботі основної системи.

Результати роботи механізму правил та аналітичних сервісів консолідуються в модулі оркестрування відповідей. Саме цей модуль приймає остаточні рішення: чи блокувати повідомлення, накладати обмеження на користувача, реєструвати інцидент, надсилати повідомлення адміністратору або запускати більш складний автоматизований сценарій. Цей модуль тісно інтегрований з панеллю адміністратора, що дозволяє системному адміністратору бачити повну картину, змінювати політику відповідей та, за необхідності, коригувати автоматичні дії.

Важливим елементом архітектури є рівень зберігання даних. Він включає оперативну реляційну базу даних (таку як PostgreSQL або MySQL), яка зберігає метадані, конфігурації, політики, інформацію про користувачів та інциденти, а також спеціалізоване сховище для журналів та часових рядів, що використовуються для аналітики та статистики. Для великих обсягів інвестицій та журналів передбачено окремий рівень зберігання файлів та архівування (холодне

зберігання), що забезпечує довгострокове зберігання даних за помірною вартістю ресурсів.

Для взаємодії з користувачами та службою безпеки створено адміністративний веб-інтерфейс. За його допомогою адміністратори, аналітики та аудитори можуть переглядати виявлені інциденти, налаштовувати політики безпеки, генерувати звіти та аналізувати динаміку порушень. Доступ до цих функцій регулюється механізмами контролю доступу на основі ролей, що дозволяють розділити повноваження між різними категоріями користувачів системи.

Нарешті, варто згадати про інтеграційний рівень, який з'єднує систему моніторингу з іншими елементами корпоративної інфраструктури. Це можуть бути системи класу SIEM, каталоги користувачів (LDAP/Active Directory), зовнішні служби аутентифікації та авторизації. Завдяки цьому результати роботи боту можуть бути використані в більш широкому контексті захисту інформаційних ресурсів організації.

Проектування архітектури базується на декількох ключових принципах. Масштабованість досягається завдяки можливості незалежно масштабувати кожну службу залежно від навантаження та використання черг повідомлень для розподілу роботи. Надійність забезпечується реплікацією бази даних, дублюванням критичних журналів та стійкістю до відмов компонентів. Безпека даних підтримується за допомогою шифрування під час зберігання та передачі, обмеженого доступу до конфіденційних об'єктів та впровадження механізмів аудиту дій системи та адміністратора. Конфігуруваність реалізується за допомогою можливості змінювати правила та політики в режимі реального часу, а прозорість досягається за допомогою ведення детальних журналів автоматичних відповідей та збереження доказів кожного інциденту. Разом це утворює архітектуру, придатну для реального використання в корпоративному середовищі з високими вимогами до інформаційної безпеки.

Узагальнену архітектуру програмної системи моніторингу корпоративних чатів у Telegram наведено на рис. 2.2.

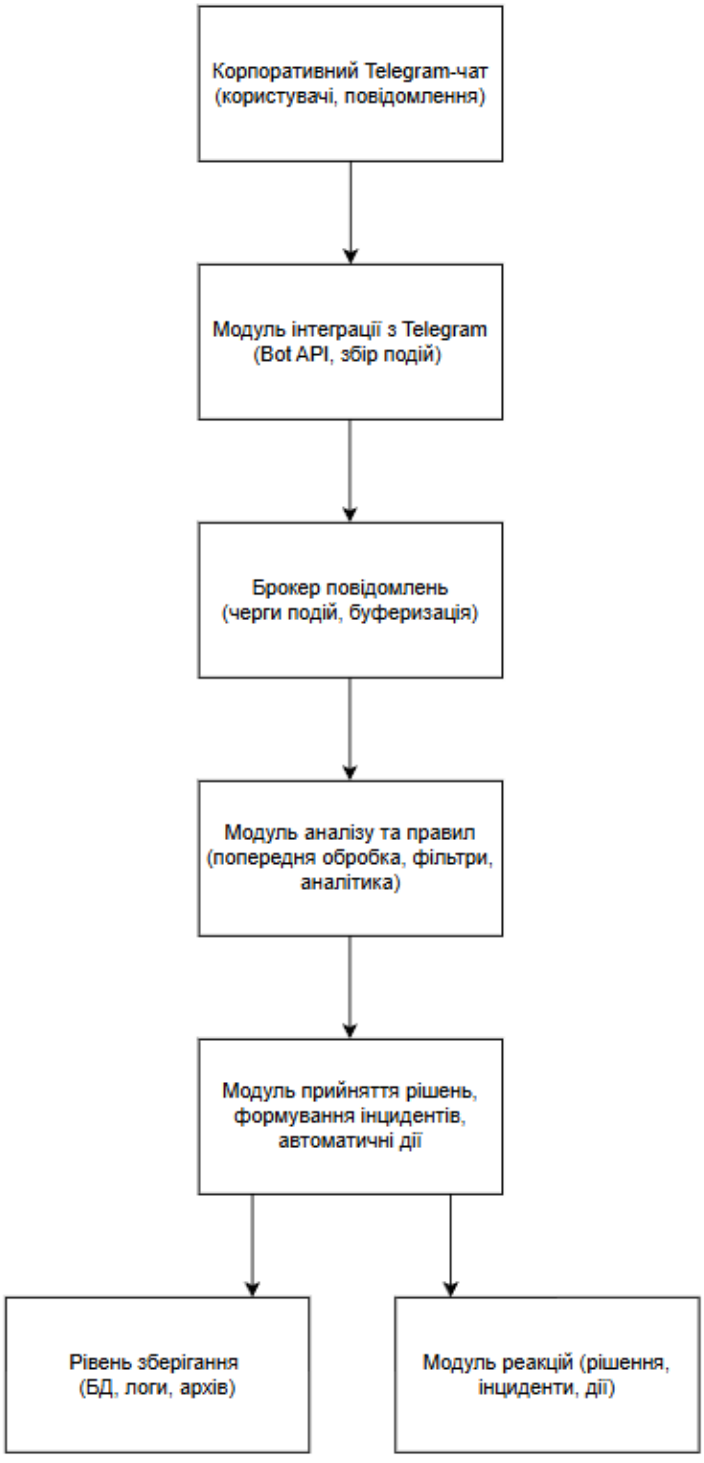


Рис. 2.2 Узагальнена архітектура програмної системи моніторингу інформаційної безпеки корпоративних чатів у Telegram

2.5 Алгоритми аналізу подій та виявлення порушень політики безпеки

Попередня фільтрація є одним з ключових етапів обробки інформаційних подій у системі моніторингу корпоративного чату Telegram. Основна мета цього процесу – швидко відфільтрувати технічно неважливі або явно безпечні повідомлення, щоб зменшити навантаження на наступні модулі глибокого аналізу. Такий підхід забезпечує високу продуктивність системи, особливо в умовах великого обсягу комунікацій, що є типовим для корпоративних середовищ.

Алгоритм попередньої фільтрації працює за принципом багатоступеневої перевірки вхідних даних, при якій на кожному етапі послідовно оцінюються повідомлення за певними критеріями. Перший рівень визначає тип контенту – текст, графіка, аудіо, відео або повідомлення сервісної системи. Така класифікація дозволяє відразу відфільтрувати повідомлення, які не можуть порушити політику безпеки, наприклад внутрішні технічні повідомлення Telegram про додавання нових учасників або зміну налаштувань чату.

Другий рівень фільтрації передбачає використання базових методів перевірки текстових повідомлень на основі правил. На цьому етапі система порівнює вміст із списком заборонених слів і фраз, набором типових маркерів ризику (номери банківських карток, паспортні дані, фінансові документи, конфіденційні внутрішні терміни) або шаблонами, які можуть вказувати на потенційно небезпечні дії. Використання регулярних виразів дозволяє швидко виявляти структуровані дані, які потенційно можуть становити загрозу, такі як номери телефонів, адреси електронної пошти та особисті ідентифікаційні номери.

Третій рівень фільтрації стосується нестандартних об'єктів, таких як документи, зображення та файли. Хоча поглиблений аналіз файлів виконується на наступних етапах, попередня фільтрація може виконувати базові перевірки: визначення типу файлу, розміру, наявності шифрування або підозрілих аномалій у метаданих. Повідомлення з файлами, що перевищують встановлені параметри безпеки, негайно позначаються як підозрілі та передаються до наступних модулів з обов'язковим пріоритетом.

Четвертий рівень передбачає оцінку моделей поведінки користувачів. Хоча аналіз на цьому етапі є поверхневим, система може виявити аномально високу частоту повідомлень, передачу великих файлів, раптові зміни в стилі спілкування або інші показники, які можуть свідчити про компрометацію облікового запису. Такі сигнали позначаються для детального аналізу в модулі поведінкової аналітики.

Попередня фільтрація також служить для оптимізації інформаційних потоків системи. Повідомлення, що не містять індикаторів ризику, можуть бути позначені як безпечні, щоб вони не надсилалися на наступні етапи, що значно зменшує обчислювальні витрати. З іншого боку, повідомлення, що пройшли хоча б один негативний критерій фільтрації, підлягають більш глибокій обробці. Завдяки такій структурі попередня фільтрація діє як «перша лінія захисту», забезпечуючи баланс між швидкістю реагування та точністю аналізу.

Ефективно реалізований алгоритм попередньої фільтрації відіграє вирішальну роль у забезпеченні масштабованості всієї системи моніторингу. Він мінімізує ресурсні витрати на обробку вторинної інформації, знижує ризик перевантаження модулів штучного інтелекту та скорочує затримку між моментом появи повідомлення та його повним аналізом. В результаті підвищується надійність і швидкість виявлення інцидентів інформаційної безпеки в корпоративних чатах Telegram.

Після проходження початкового фільтрування повідомлення надсилається до модуля детального аналізу, який виконує поглиблену перевірку текстового вмісту за допомогою лінгвістичних і статистичних методів. Основна мета цього етапу – виявити приховані ризики, які неможливо виявити простим порівнянням повідомлення зі списками заборонених ключових слів. Сучасні текстові дані часто містять непрямі посилання на конфіденційну інформацію, абревіатури, синоніми або контекстні натяки, тому використання технологій NLP забезпечує значно вищу точність.

Узагальнений алгоритм обробки подій у системі моніторингу корпоративного чату Telegram наведено на рис. 2.3.

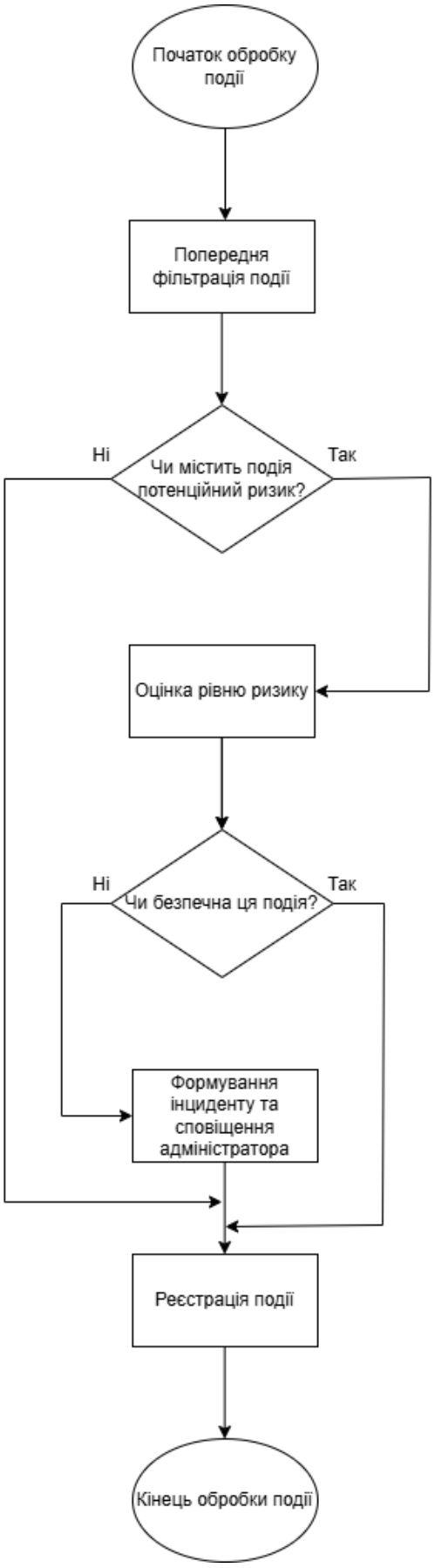


Рис. 2.3 Узагальнений алгоритм обробки подій та виявлення порушень політик безпеки в системі моніторингу корпоративного чату Telegram

Алгоритм складається з декількох етапів. На першому етапі текст нормалізується: токенізація, переведення в нижній регістр, видалення стоп-слів, лематизація або стемінг. Це дозволяє уніфікувати дані та підготувати їх до аналізу. Далі система застосовує контекстну перевірку, під час якої повідомлення оцінюються на відповідність політикам безпеки за допомогою словників та тематичного аналізу. Наприклад, заборонене слово може не становити ризику, якщо воно використовується в технічному або освітньому контексті.

Третій етап передбачає використання моделей машинного навчання, таких як текстові класифікатори (Naïve Bayes, SVM, Random Forest) або сучасні трансформерні моделі (BERT, RoBERTa). Ці моделі здатні визначати рівень ризику повідомлення на основі контексту, структури речень, стилістичних маркерів та семантичних патернів. У корпоративному середовищі такі алгоритми дозволяють, наприклад, виявляти повідомлення з ознаками соціальної інженерії або прихованої передачі конфіденційних даних.

Остання частина алгоритму передбачає формування оцінки ризику. Повідомлення отримує числовий показник, який враховує як лінгвістичні патерни, так і загальні правила політики безпеки. Якщо оцінка перевищує встановлений поріг, повідомлення передається до модуля формування інциденту. Таким чином, алгоритм детального аналізу тексту забезпечує точну та контекстно-залежну оцінку комунікацій у корпоративному чаті.

Аналіз поведінки є ключовим елементом сучасних систем моніторингу інформаційної безпеки, оскільки значна частина інцидентів з інсайдерами безпосередньо пов'язана не з вмістом повідомлення, а з незвичайною активністю користувача. У корпоративних чатах Telegram аномалії можуть включати різке збільшення активності, нетипові інтервали часу взаємодії, масові передачі файлів, зміну тону спілкування або дії, нехарактерні для ролі співробітника.

Алгоритм поведінкового аналізу працює шляхом збору статистичних даних про звичайний профіль активності кожного користувача. На першому етапі формується базова модель поведінки, яка включає частоту повідомлень, середній обсяг вмісту, типові години спілкування, характерний набір чатів і типовий стиль

мовлення. Ця модель зберігається як «норма» і використовується для порівняння з поточною активністю.

Другий етап – аналіз відхилень від норми. Тут використовуються алгоритми виявлення аномалій: Isolation Forest, DBSCAN, One-Class SVM або методи кластеризації. Вони дозволяють виявляти поведінку, що виходить за межі типових шаблонів. Наприклад, співробітник, який раніше надсилав в середньому 5-7 файлів на день на адресу , раптом починає надсилати десятки документів за короткий проміжок часу, що може свідчити про спробу витоку інформації.

Третій етап враховує контекстні метадані: нове підключення з нетипової IP-адреси, активність у незвичайний час доби, різка зміна стилю повідомлень або підозрілі зміни в журналах чату (видалення старих повідомлень, редагування нового вмісту). Разом ці показники формують поведінковий профіль ризику.

На четвертому етапі система обчислює інтегральний показник аномалії. Якщо значення перевищує поріг безпеки, активність позначається як потенційний інцидент і передається до модуля генерації попереджень. Цей алгоритм дозволяє виявляти складні загрози, які неможливо ідентифікувати лише на основі змісту повідомлень.

У корпоративних чатах Telegram значна частина переданої інформації міститься не в текстових повідомленнях, а в файлах – документах, зображеннях, архівах, таблицях та інших матеріалах. Тому ефективний аналіз вкладень є важливою складовою системи моніторингу безпеки. Основна мета алгоритму – виявлення файлів, які можуть становити загрозу витоку інформації або містити потенційно небезпечний вміст.

Алгоритм починається з первинної ідентифікації файлу: визначення його типу, розміру, формату, хеш-суми та основних метаданих. На цьому етапі файли, які явно перевищують вимоги політики безпеки за розміром, не відповідають дозволеним форматам або мають підозрілі структурні характеристики, можуть бути відразу відкинуті.

Другий етап – структурний аналіз. Документи (PDF, DOCX, XLSX) можуть містити метадані, що розкривають внутрішню інформацію, таку як ім'я автора,

назва організації, шляхи зберігання файлів тощо. Для графічних файлів аналізуються дані EXIF, які можуть містити GPS-координати, назву пристрою або дату створення. Вся ця інформація оцінюється на відповідність політикам безпеки.

Третій етап передбачає аналіз вмісту. Для текстових документів текст витягується і перевіряється на наявність заборонених фраз, структурованих конфіденційних даних, фінансової та особистої інформації. Для зображень можуть використовуватися алгоритми OCR для виявлення тексту, прихованого на фотографіях документів.

Четвертий етап – антивірусне сканування та сканування сигнатур. Це особливо важливо для архівів та виконуваних файлів, щоб запобігти поширенню шкідливого програмного забезпечення через корпоративні канали.

На останньому етапі файл отримує інтегровану оцінку ризику. Система визначає, чи відповідає його передача політикам безпеки, і генерує інцидент, якщо виявляє порушення. Цей алгоритм забезпечує комплексний контроль над усіма типами вкладень.

Після виявлення системою порушення політики або ненормальної активності інцидент повинен бути правильно класифікований і для адміністратора має бути згенеровано інформаційне повідомлення. Алгоритм класифікації інцидентів виконує функцію систематизації, визначення рівня критичності та ініціювання відповідних дій.

На першому етапі інцидент класифікується за типом: витік даних, передача забороненої інформації, підозріла активність користувача, порушення політики файлів, поведінкові аномалії, технічні порушення тощо. Таке групування дозволяє адміністраторам швидко зрозуміти характер події.

Другий етап передбачає визначення рівня критичності. Це робиться за допомогою таких критеріїв, як тип даних, що потенційно могли бути розкриті, масштаб інциденту, роль користувача, обсяг повідомлень, частота відхилень від норми та можливі наслідки. Критичність може варіюватися від низької до високої.

Третій етап передбачає створення повідомлення про безпеку. Воно повинно бути лаконічним і містити ключові факти, час, тип інциденту, користувача, зміст

повідомлення або файлу (у разі дозволених політик) та рекомендації щодо подальших дій. Система може передбачати автоматичне створення детального звіту для журналу аудиту.

На четвертому етапі повідомлення передається через механізм сповіщення: у приватний чат адміністратора, канал безпеки або зовнішню систему SIEM.

Цей алгоритм завершує повний цикл обробки інцидентів і забезпечує ефективне реагування.

2.6 Проектування структури бази даних та інформаційних потоків

Проектування структури бази даних та організація інформаційних потоків є одним з ключових етапів побудови системи моніторингу інформаційної безпеки корпоративних чатів у Telegram, оскільки саме на цьому рівні визначається, як система буде зберігати, пов'язувати та аналізувати дані, що надходять із робочого середовища. Швидкість доступу до інформації, здатність обробляти великі обсяги повідомлень, підтримка історії інцидентів, виконання складних аналітичних запитів і загальна стабільність системи при зростаючому навантаженні залежать від того, наскільки добре побудована модель даних.

Системи моніторингу характеризуються тим, що дані генеруються майже безперервно і мають неоднорідну структуру. Частина з них добре структурована – ідентифікатори користувачів, часові мітки подій, типи контенту, індикатори ризику. Решта – погано формалізована: тексти повідомлень, імена файлів, результати семантичного аналізу, коментарі адміністраторів. До цього додаються події сервісу (вхід/вихід з чату, зміна прав доступу), записи інцидентів, профілі поведінки користувачів і системні журнали. Все це вимагає гнучкої, але чітко структурованої архітектури бази даних, яка може працювати як зі структурованою, так і з напівструктурованою інформацією.

Перед побудовою логічної моделі були сформульовані загальні вимоги до підсистеми зберігання. З точки зору функціональності база даних повинна

забезпечувати збереження всіх типів подій з чату Telegram – текстових повідомлень, вкладень, фотографій, посилань, сервісних дій – разом з детальними метаданими про час, відправника, чат і тип контенту.

З огляду на ці критерії, доцільно використовувати реляційну модель з можливістю подальшого розширення, наприклад, на основі PostgreSQL. При необхідності її можна доповнити гібридною архітектурою, де основні структуровані дані зберігаються в схемі SQL, а повнотекстовий пошук виконується за допомогою спеціалізованих сховищ NoSQL. В рамках даної роботи основна увага приділяється реляційній моделі, оскільки вона забезпечує транзакційність, цілісність, зрозумілі засоби моделювання відносин і добре підходить для побудови систем, орієнтованих на інциденти.

На концептуальному рівні модель даних включає ряд сутностей, які представляють основні об'єкти предметної області. До них відносяться користувачі чату, текстові повідомлення, вкладення, події сервісу, зареєстровані інциденти, політики безпеки, результати аналітичного модуля, поведінкові профілі та системні журнали. Цей набір сутностей дозволяє реалізувати повний цикл управління інцидентами – від моменту появи повідомлення в чаті до реєстрації порушення, прийняття рішення адміністратором і подальшого аналізу.

Таблиця, що описує користувачів (умовно Users), зберігає основну інформацію про учасників корпоративного чату: унікальний Telegram ID, відображуване ім'я та псевдонім, роль у чаті (звичайний учасник або адміністратор), а також часові мітки першого появи та останньої активності. Це дозволяє не тільки ідентифікувати автора кожного повідомлення, але й побудувати його поведінковий профіль – відстежувати частоту надсилання повідомлень, типові години активності, схильність до передачі файлів тощо.

Однією з найбільших таблиць за обсягом є таблиця «Повідомлення». Вона містить внутрішній ідентифікатор запису, посилання на ідентифікатор Telegram повідомлення, ключ користувача автора, текст повідомлення, тип вмісту (звичайний текст, медіа, сервісне повідомлення), час відправлення та вказівку на наявність вкладення. Ця таблиця є основним джерелом даних для модулів аналізу,

а також ядром, до якого «прив'язані» інші сутності – файли, результати аналізу, інциденти.

Концептуальну ER-діаграму бази даних розробленої системи моніторингу інформаційної безпеки корпоративних чатів у Telegram наведено на рис. 2.4.

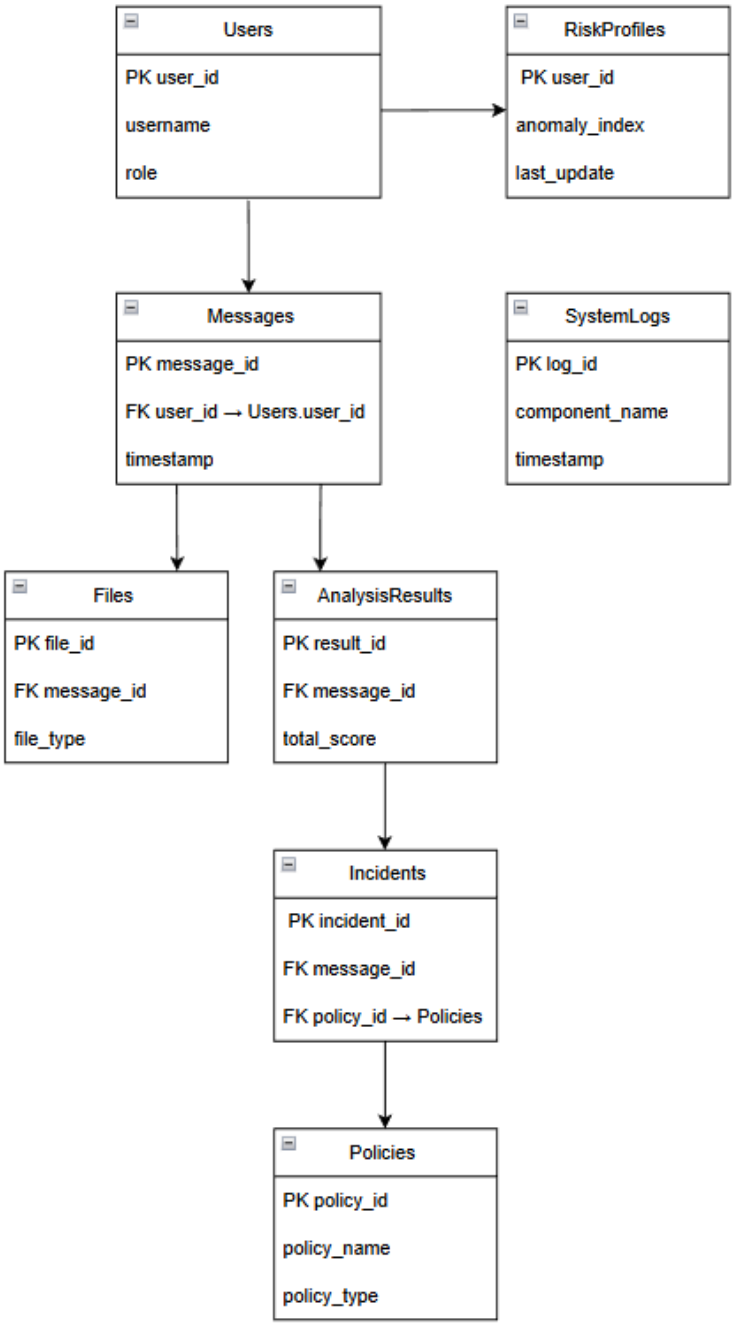


Рис. 2.4 Концептуальна модель бази даних системи моніторингу інформаційної безпеки корпоративних чатів у Telegram

Вкладення зберігаються в окремій таблиці (Файли), де кожен файл пов'язаний з оригінальним повідомленням, його оригінальною назвою, типом,

розміром, контрольною сумою та потенційним прапорцем конфіденційності. Такий підхід дозволяє аналізувати передачу файлів незалежно від текстового вмісту: наприклад, складати статистику за типами файлів, шукати нетипові піки в обсязі переданої інформації або відстежувати повторне надсилання одного і того ж документа різним користувачам.

Таблиця Інциденти займає центральне місце в моделі даних. Вона відображає кожен випадок, коли система визнала певну подію підозрілою або такою, що порушує політику безпеки. Для кожного інциденту зберігається посилання на повідомлення та автора, рівень ризику, політику, що спрацювала, опис, час виникнення та поточний статус (новий, у процесі, закритий). Це дозволяє побудувати повний життєвий цикл інциденту та організувати роботу адміністратора за принципами черги завдань.

Додаткові таблиці підтримують роботу аналітичних модулів. Наприклад, AnalysisResults записує оцінки, отримані за допомогою різних алгоритмів – сигнатурних, евристичних, поведінкових, семантичних – та їх агреговану оцінку ризику. Таблиця RiskProfiles підсумовує показники поведінки користувачів: середню кількість повідомлень на день, типові години активності, середню кількість переданих файлів, індекс аномалій та час останнього оновлення профілю. Таблиця Policies містить формалізоване опис правил безпеки – ідентифікатор, назва, тип політики, текстове пояснення, статус активності. Нарешті, SystemLogs відображає внутрішні події самої системи: роботу окремих компонентів, помилки, перезапуски, зміни конфігурації. Це створює основу для технічного аудиту та усунення можливих несправностей.

Не менш важливим, ніж логічна структура, є питання організації інформаційних потоків. Зовні система отримує дані безпосередньо з Telegram: текстові повідомлення, вкладення, події сервісу та інформацію про користувачів. Все це проходить через модуль інтеграції на рівень збору та нормалізації, де події перетворюються на уніфіковані записи, які потім надсилаються до аналітичних модулів. Після обробки приймається рішення щодо наявності порушення; у разі необхідності створюється запис у таблиці інцидентів, оновлюються результати

аналізу та змінюються профілі поведінки користувачів. На основі цих даних формуються звіти для адміністратора та надсилаються повідомлення про критичні випадки.

Оскільки система працює з потенційно конфіденційною інформацією, механізми захисту даних у базі даних мають особливе значення. На рівні схеми доступ до таблиць диференціюється за ролями – розділяючи права боту, адміністратора безпеки та аналітика. Для чутливих полів може використовуватися шифрування, а з'єднання з базою даних захищаються за допомогою TLS. Крім того, реалізовано резервне копіювання, реплікацію критичних даних для підвищення відмовостійкості та контроль цілісності за допомогою хеш-функцій або цифрових підписів.

Для підтримки масштабованості база даних повинна бути готова до зростання кількості користувачів і обсягу історичних записів. Для цього до часто використовуваних полів (ідентифікатор повідомлення, час відправлення, ідентифікатор користувача) застосовуються індекси, таблиці горизонтально розділяються за часом або логічними критеріями, типові запити кешуються, а транзакційні та аналітичні навантаження розділяються. Типовим рішенням є використання основної бази даних для операційних транзакцій, репліки для аналітики та звітності, а також окремого архівного сховища для довгострокового зберігання інцидентів та журналів. Така організація дозволяє уникнути перевантаження основної бази даних, зберігаючи при цьому можливість повного відновлення історії подій у будь-який момент часу.

В результаті запропонована структура бази даних і організація інформаційних потоків забезпечують цілісність, керованість і масштабованість системи моніторингу . Вони дозволяють ефективно накопичувати і аналізувати дані, підтримують складні алгоритми виявлення вторгнень і надають адміністраторам всю необхідну інформацію для прийняття рішень в області інформаційної безпеки.

Висновки до розділу 2

У другому розділі наведено теоретичне обґрунтування, моделювання та проектування системи моніторингу інформаційної безпеки корпоративних чатів у Telegram. Розглянуто основні концепції моніторингу, міжнародні стандарти (ISO/IEC 27001, NIST тощо), принципи Zero Trust та SOC, а також показано необхідність їх адаптації до специфіки месенджерів, які характеризуються високою динамікою інформаційних потоків і відсутністю вбудованого централізованого контролю.

Проаналізовано наявні моделі виявлення інцидентів та оцінки ризиків, зокрема сигнатурні, евристичні, поведінкові та семантичні підходи, а також класичні моделі STRIDE, DREAD, OCTAVE. На основі цього побудовано концептуальну модель системи моніторингу та розроблено архітектуру, що включає рівні збору даних, аналітики, прийняття рішень і зберігання. Описано основні алгоритмічні підходи до аналізу подій, серед яких важливе місце посідають методи обробки природної мови (NLP) для інтерпретації змісту повідомлень.

Окремо спроектовано структуру бази даних та інформаційних потоків, визначено ключові сутності (користувачі, повідомлення, файли, інциденти, профілі ризику, політики, журнали) та їх взаємозв'язки. Показано, що запропонована модель забезпечує можливість довгострокового зберігання даних, виконання складних аналітичних запитів і масштабування системи.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ МОНІТОРИНГУ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ КОРПОРАТИВНИХ ЧАТІВ У TELEGRAM

3.1 Загальні принципи реалізації системи та вибір програмних засобів

Проектування системи моніторингу інформаційної безпеки корпоративних чатів у Telegram вимагає комплексного підходу, що охоплює як технічні аспекти, так і організаційні, процедурні та поведінкові фактори. З огляду на те, що сучасні корпоративні комунікації характеризуються високою швидкістю обміну даними, низьким рівнем формалізації та широким спектром сценаріїв взаємодії між співробітниками, система моніторингу повинна відповідати ряду фундаментальних вимог. Перш за все, важливо забезпечити гнучкість, яка дозволяє адаптувати механізми виявлення порушень до різних моделей поведінки користувачів, особливостей бізнес-процесів та рівня ризику в конкретній організації.

Система повинна бути здатна працювати в умовах значної невизначеності, коли потенційна загроза може проявлятися як у вигляді прямого порушення політики безпеки, так і через непрямі сигнали, що вимагають контекстного аналізу. Наприклад, передача файлів може бути як законною робочою діяльністю, так і ознакою витоку інформації. Саме тому дизайн системи базується на багаторівневій архітектурі, в якій кожен рівень відповідає за свій сегмент обробки даних – від початкового перехоплення повідомлень до їх високорівневої інтерпретації та оцінки ризиків.

При створенні системи також важливо враховувати вимоги до її невидимості та непомітності для користувачів. Вона не повинна уповільнювати корпоративний чат, створювати затримки або викликати помилкові спрацьовування, які заважатимуть комунікації. Оптимальний підхід – це коли користувач повністю не помічає систему моніторингу, за винятком випадків, коли вона блокує потенційно небезпечні дії або надсилає попередження про порушення політики.

Особливу увагу слід приділити вимогам безпеки самої системи, яка не повинна стати новою точкою вразливості. Механізм отримання даних через Telegram Bot API повинен бути безпечним, а доступ до бази даних – суворо регульованим. Усі операції, пов'язані з обробкою конфіденційної інформації, повинні виконуватися в безпечному середовищі, що мінімізує ризики витоку або несанкціонованої модифікації записів.

Серед технічних вимог важливо виділити продуктивність і масштабованість. Оскільки корпоративні чати Telegram можуть містити десятки або навіть сотні активних учасників, кількість повідомлень за короткий проміжок часу може бути значною. Система повинна бути здатна обробляти великий потік даних без зниження швидкості відгуку.

Важливим принципом є модульність, яка дозволяє замінювати або оновлювати окремі частини системи без зупинки її роботи. Такий підхід не тільки забезпечує гнучкість, але й можливість інтегрувати нові алгоритми, такі як моделі машинного навчання для кращого розпізнавання ризикованих ситуацій. Модульність також сприяє швидшому виправленню помилок і спрощує обслуговування програмного забезпечення.

З огляду на вищезазначене, загальні принципи проектування системи можна підсумувати як прагнення забезпечити баланс між точним виявленням інцидентів, мінімальним впливом на користувальницький досвід та високим рівнем безпеки, надійності та масштабованості. Ці підходи є основою для подальшого проектування та впровадження системи моніторингу інформаційної безпеки корпоративних чатів у Telegram.

3.2 Практична реалізація архітектури програмного забезпечення системи моніторингу

Ефективне проектування системи моніторингу інформаційної безпеки корпоративних чатів у Telegram неможливе без ретельного аналізу вимог, що

базуються на специфіці корпоративного середовища, нормативних обмеженнях, технічних можливостях платформи та очікуваннях користувачів. Цей розділ має на меті сформувати чітке розуміння того, що саме повинна забезпечувати система, які функції вважаються критично необхідними та які критерії визначають рівень її успішності. Щоб створити дійсно ефективне рішення, вимоги слід розглядати не як абстрактні побажання, а як конкретні умови, без яких система втрачає свою практичну цінність.

Однією з ключових вимог є забезпечення безперервного моніторингу комунікацій у корпоративному чаті Telegram. Система повинна працювати стабільно і безперервно, аналізуючи кожне отримане повідомлення незалежно від часу доби та рівня навантаження. У сучасних цифрових бізнес-процесах навіть короткочасний збій або пропущені події можуть становити значний ризик витоку інформації. Тому надзвичайно важливо, щоб система мала високий рівень доступності та стійкості до збоїв. Ця вимога передбачає використання механізмів резервного копіювання, систем контролю доступності та можливості швидкого відновлення після надзвичайних ситуацій.

Іншим важливим аспектом є точність виявлення інцидентів. Система повинна не тільки перехоплювати повідомлення, але й правильно інтерпретувати їхній зміст, відокремлюючи реальні загрози від безпечного спілкування. Надмірна кількість помилкових спрацьовувань призводить до втрати довіри до інструменту, перевантажує адміністратора повідомленнями і, в кінцевому підсумку, призводить до ігнорування реальних загроз. Саме тому критерії точності та ефективності виявлення порушень мають центральне значення. Це вимагає використання сучасних методів семантичного аналізу тексту, виявлення аномалій та оцінки контексту повідомлень.

Важливу роль відіграє також вимога мінімального впливу на користувачів. Система повинна працювати у фоновому режимі, не заважаючи комунікації. Вона не повинна змінювати повідомлення, втручатися в обговорення або уповільнювати чат. Будь-які попередження, що надсилаються користувачам, повинні бути сформульовані таким чином, щоб не паралізувати робочі процеси та не створювати

атмосферу надмірного контролю. Баланс між безпекою та конфіденційністю комунікації є важливою складовою системних вимог, тому повинен надавати механізми, що дозволяють гнучко налаштовувати рівень контролю залежно від політики конкретної організації.

Не менш важливим є питання масштабованості. Оскільки обсяг трафіку в корпоративних чатах може значно варіюватися залежно від кількості співробітників, часу доби або інтенсивності бізнес-процесів, система повинна бути здатна витримувати такі коливання. Вона повинна підтримувати збільшення кількості користувачів, підключення додаткових груп або каналів, а також розширення функціональності без необхідності повного перепроектування. Масштабованість стосується як технічної сторони – продуктивності сервера, так і логічної – структури даних, архітектурних рішень і форматів логування.

Окремо варто зупинитися на вимозі щодо забезпечення відповідності нормативним та внутрішнім корпоративним правилам. Багато організацій, особливо великі компанії, повинні дотримуватися нормативних вимог у сфері захисту персональних даних, конфіденційної інформації та комерційної таємниці. Тому система моніторингу повинна не тільки виявляти інциденти, але й гарантувати, що сам процес аналізу даних не порушує правових норм. Наприклад, важливо забезпечити доступ до даних тільки уповноваженим особам, використовувати сучасні методи шифрування та не зберігати інформацію довше, ніж того вимагає політика безпеки.

Що стосується критеріїв ефективності, то вони включають не тільки технічні показники, такі як швидкість обробки, кількість оброблених повідомлень або рівень помилкових спрацьовувань, але й якісні характеристики. До них належить здатність системи генерувати корисну аналітичну інформацію, яка допомагає адміністраторам приймати обґрунтовані рішення. Іншими словами, система повинна не просто повідомляти про інциденти, а надавати адміністраторам повну картину контексту, тенденцій та потенційних наслідків порушень. Саме така інтегрована аналітика перетворює систему з простого інструменту сповіщення на стратегічно важливий елемент інформаційної безпеки.

Важливо зазначити, що системні вимоги не є статичними, а динамічними. Вони залежать від розвитку корпоративної культури, впровадження нових інструментів, змін у зовнішніх загрозах та появи нових типів інцидентів. Тому система повинна бути здатна адаптуватися до змін як на рівні конфігурації, так і на рівні архітектури. Саме динамічний характер вимог визначає необхідність створення гнучкої моделі, яка дозволяє легко впроваджувати оновлення, додаткові модулі, нові алгоритми або форми звітності.

Підсумовуючи, аналіз вимог до системи моніторингу інформаційної безпеки корпоративного чату в Telegram дозволяє сформувати комплексний набір характеристик, без яких її функціонування буде неповним або неефективним. Підтримка точного аналізу, висока продуктивність, безперебійність роботи, відповідність нормативним вимогам та гнучка структура є ключовими критеріями, що визначають ефективність системи та формують основу для подальших етапів її проектування та впровадження.

3.3 Реалізація Telegram-боту та функціональних модулів моніторингу

Розробка ефективної системи моніторингу інформаційної безпеки корпоративних чатів передбачає створення концептуальної моделі, яка визначає логічну структуру програмного комплексу, функції його основних компонентів та характер їх взаємодії. Така модель дозволяє сформувати цілісне уявлення про всі процеси, що відбуваються в системі, від моменту надходження повідомлення в корпоративний чат до його аналізу, реєстрації інциденту та повідомлення відповідальних осіб. Без побудови концептуальної моделі неможливо забезпечити координацію компонентів, передбачити потенційні сценарії ризиків та забезпечити масштабованість системи в майбутньому.

Telegram як корпоративна комунікаційна платформа надає широкий спектр можливостей інтеграції, але водночас накладає певні обмеження на сторонні системи моніторингу. Повідомлення, файли, системні події та метадані надходять

у вигляді потоків, обсяг яких може значно варіюватися залежно від активності користувачів. Система повинна бути здатна перехоплювати ці потоки в режимі реального часу, аналізувати їхній зміст і контекст та надавати адміністраторам узагальнені оцінки безпеки. Тому концептуальна модель повинна відображати багаторівневу обробку, від початкового збору даних до поглибленої оцінки ризиків.

У рамках цієї моделі всі компоненти системи умовно поділяються на кілька логічних сегментів. Перший сегмент відповідає за взаємодію з API Telegram, забезпечуючи прийом і передачу даних. Другий сегмент виконує попередню нормалізацію повідомлень, дозволяючи привести їх до єдиної структури для подальшої обробки. Третій сегмент реалізує аналітичні механізми, що включають семантичний аналіз тексту, виявлення прихованих ризиків, аналіз вкладень і спроби небажаної активності. Четвертий сегмент пов'язаний з прийняттям рішень – це центральний елемент, який визначає реакцію системи на конкретну подію. Останній сегмент виконує функції зберігання даних, ведення журналу та звітності.

Однією з ключових особливостей концептуальної моделі є її фокус не тільки на технічних аспектах обробки даних, але й на організаційному контексті. У корпоративних чатах генерується велика кількість неформальних повідомлень, які можуть містити як інформацію, пов'язану з роботою, так і емоційно заряджені реакції, дискусії, жарти або внутрішні домовленості. Через це система повинна вміти відрізняти загрози від безпечного, нормального спілкування. Саме тому алгоритми контекстного аналізу та поведінкові профілі відіграють важливу роль у концептуальній моделі, дозволяючи визначати не тільки зміст повідомлень, але й характер спілкування між учасниками.

Першим елементом концептуальної моделі є рівень, відповідальний за пряму взаємодію системи з Telegram. Саме на цьому рівні формується первинний потік даних, який є джерелом подальшої аналітики. Telegram Bot API надає розробнику можливість отримувати текстові повідомлення, документи, фотографії, голосові повідомлення, системні події, інформацію про зміни статусу учасників та інші типи даних. Важливо, щоб система приймала цей потік без затримок і гарантувала, що жодне повідомлення не буде втрачено, навіть під час пікових навантажень.

Для обробки подій в режимі реального часу зазвичай використовується механізм Webhook, який забезпечує отримання інформації відразу після її появи в чаті. Тому рівень інтеграції повинен бути максимально стабільним, з мінімальною затримкою і чітко визначеною логікою перенаправлення повідомлень на наступні компоненти. В іншому випадку система ризикує втратити частину інформації або сформувати неправильні висновки.

На цьому рівні отримані дані також нормалізуються. Повідомлення можуть відрізнятися за форматом, мовою, структурою або містити вкладення різних типів. Тому завданням рівня інтеграції є приведення всіх повідомлень до єдиного формату, який можна ефективно передати на аналітичний рівень. Сюди входить очищення тексту, відокремлення метаданих, ідентифікація типу події та попередній аналіз для визначення, які модулі повинні далі обробляти цей тип даних.

Аналітичний рівень є центральною частиною концептуальної моделі, оскільки саме тут відбувається найглибший аналіз повідомлень, вкладень та поведінки користувачів. На цьому рівні розпізнаються ризиковані дії, шукаються приховані загрози та виявляються потенційні порушення політики безпеки. Аналітика охоплює як текстовий вміст, так і файли, зображення та інші нестандартні формати даних.

Система повинна бути здатна аналізувати текст не тільки на наявність заборонених слів, але й оцінювати його семантичний зміст. Наприклад, слово «пароль» може використовуватися в контексті попередження або навчання, але воно також може сигналізувати про спробу передачі конфіденційної інформації. Ці відмінності можна визначити лише за допомогою складних алгоритмів, що враховують контекст. Саме тому аналітичний рівень включає модулі NLP, які дозволяють визначити тон повідомлення, його зміст та можливу мотивацію відправника.

Файли та вкладення розглядаються окремо. Система повинна виявляти несанкціоновану передачу документів, які можуть містити конфіденційну інформацію. Для цього необхідний аналіз імен файлів, метаданих, розмірів і, за

необхідності, аналіз змісту за допомогою технологій оптичного розпізнавання символів або баз даних підписів.

Аналіз поведінки є окремою областю аналітики. Деякі порушення виявляються не за допомогою конкретного повідомлення, а за допомогою зміни активності користувача. Наприклад, різке збільшення кількості повідомлень вночі або передача великої кількості файлів за короткий проміжок часу може вказувати на потенційний інцидент. Тому система повинна генерувати поведінкові профілі, які дозволяють оцінювати дії користувачів не ізольовано, а в більш широкому контексті.

В рамках концептуальної моделі одним з ключових компонентів є модуль оцінки ризиків, який визначає ступінь загрози, яку становить отримане повідомлення або подія для інформаційної безпеки підприємства. Цей модуль є центральним ланцюгом між аналітичним рівнем, який збирає та попередньо обробляє дані, і механізмами реагування, які визначають конкретні дії системи у відповідь на інцидент. Основна роль модуля полягає в узагальненні результатів аналізу, інтерпретації виявлених закономірностей та формуванні оцінки ризику, яка використовується для подальших управлінських рішень.

Процес оцінки ризиків не можна звести до простого пошуку ключових слів або закономірностей – такий підхід у корпоративному середовищі є недостатньо точним і призводить до великої кількості помилкових спрацьовувань. Саме тому модуль прийняття рішень враховує не тільки зміст повідомлень, але й широкий спектр додаткових факторів: історію поведінки користувача, характер його попередньої діяльності, час і контекст події, наявність вкладень, тип взаємодії між учасниками чату та відповідність повідомлення корпоративним політикам. Така складність дозволяє більш реалістично оцінювати загрози та мінімізувати ймовірність помилкових спрацьовувань.

Оцінка ризиків починається з агрегації всіх параметрів, отриманих на аналітичному рівні. Характеристики тексту, семантичні особливості, параметри файлів, метадані та індикатори поведінкових моделей формують базовий профіль події. Потім система співвідносить ці дані з правилами безпеки, що діють в

конкретній організації. Наприклад, в деяких компаніях передача файлів на поза робочим часом вважається нормальною, а в інших це може вважатися високим ризиком. Ось чому модуль оцінки ризиків повинен бути гнучким і адаптуватися до корпоративних політик.

Після аналізу система генерує числову або категоріальну оцінку ризику. У найпростішому випадку це може бути трирівнева шкала: низький, середній і високий ризик. У більш складних сценаріях використовується багатовимірна модель, яка дозволяє враховувати взаємодію різних факторів. Наприклад, повідомлення, яке саме по собі не містить конфіденційних даних, але було надіслано користувачем з історією підозрілої активності, може бути класифіковано як ризикове. І навпаки, повідомлення з технічно ризикованим вмістом від користувача, який постійно демонструє безпечну поведінку, може отримати набагато нижчий пріоритет.

Після формування оцінки ризику система повинна вирішити, як реагувати. У разі низького рівня ризику подія може бути лише записана в системному журналі для подальшого аналізу. Якщо ризик середній, система може надіслати повідомлення адміністратору, який може самостійно оцінити ситуацію. У разі високого ризику система може автоматично заблокувати файл або повідомлення та вжити додаткових заходів, залежно від налаштувань корпоративної політики. Цей механізм дозволяє збалансувати автоматизацію та ручне управління, що є оптимальним підходом у сфері інформаційної безпеки.

Важливо також зазначити, що модуль оцінки ризиків не є статичним. Система повинна вміти вчитися на попередніх інцидентах, коригувати коефіцієнти ризику та адаптуватися до нових типів загроз. Це робить її живим елементом корпоративної інфраструктури, здатним реагувати на зміни в поведінці користувачів або появу нових векторів атак. У цьому сенсі модуль оцінки ризиків є не просто інструментом аналізу, а центром прийняття рішень, який відіграє ключову роль у забезпеченні інформаційної безпеки.

Розробка концептуальної моделі неможлива без детального розуміння того, як система повинна зберігати дані. Логування є одним з найважливіших аспектів

системи моніторингу, оскільки саме через журнал подій документується вся активність у корпоративному чаті, включаючи повідомлення, інциденти, порушення та відповіді адміністратора. Якість логування визначає не тільки можливість аналізу минулих подій, але й здатність організації проводити аудит інформаційної безпеки та оцінювати ефективність політик.

Система повинна зберігати дані в структурованій формі, що дозволяє здійснювати пошук, фільтрацію та аналіз інформації за різними параметрами. Зберігання текстових повідомлень, файлів, інформації про користувачів, часових міток та рівнів ризику подій повинно бути організовано таким чином, щоб адміністратор міг швидко отримати доступ до необхідних даних. Крім того, важливо забезпечити належне управління правами доступу. Інформація, що міститься в журналі, може бути конфіденційною, тому доступ до неї повинен бути суворо регламентований.

Ще одним важливим завданням логування є забезпечення незмінності даних. Умова незмінності є критично важливою, оскільки будь-яка модифікація історичних записів може призвести до спотворення результатів аналізу або фальсифікації інформації. Тому система повинна мати механізми, що запобігають модифікації або видаленню записів без належного дозволу. У деяких випадках можуть навіть використовуватися технології незмінних журналів або криптографічні підписи.

Крім зберігання даних, цей рівень також відповідає за створення резервних копій і забезпечення відновлення інформації в разі збоїв. Втрата журналів може мати серйозні наслідки, тому система повинна мати засоби для автоматичного резервного копіювання, які запускаються через певні проміжки часу або після досягнення певного обсягу даних. Це забезпечує цілісність історії подій навіть у разі форс-мажорних обставин.

Останнім елементом концептуальної моделі є модуль взаємодії адміністратора, який не тільки надає доступ до журналів подій і результатів аналізу, але й генерує чітке тлумачення даних. Саме цей модуль перетворює численні технічні сигнали, сповіщення та оцінки ризиків у узагальнену

інформацію, яка має практичну цінність для осіб, відповідальних за інформаційну безпеку.

У корпоративному середовищі адміністратори рідко мають можливість обробляти тисячі повідомлень або брати участь у кожній дискусії. Тому система повинна подавати інформацію у зручній текстовій формі, яка дозволяє швидко оцінити ситуацію – визначити, чи відбувся реальний інцидент, які користувачі демонструють підозрілу активність, як загальний рівень ризику змінюється з часом, які теми викликають найбільше порушень тощо. Вся ця інформація повинна бути інтегрована в цілісний, логічно структурований опис, який легше зрозуміти, ніж необроблені дані.

Механізм подання результатів повинен враховувати як контекст конкретної події, так і загальну картину. Наприклад, якщо один користувач неодноразово порушує політику безпеки, адміністратор може визначити, чи є це результатом недостатньої обізнаності або навмисних дій. Система повинна допомагати виявляти такі закономірності та надавати додаткові пояснення ймовірних причин інцидентів. У довгостроковій перспективі це дозволяє не тільки реагувати на загрози, але й розробляти превентивні заходи, розробляти програми навчання та оптимізувати корпоративну політику.

Окрім текстового представлення, важливим компонентом є генерація статистичних даних, що дозволяє відстежувати динаміку порушень. Система має можливість узагальнювати інформацію про інциденти за різними критеріями, такими як періоди часу, групи користувачів, рівні ризику, типи виявлених порушень або канали комунікації. Це дозволяє виявляти проблемні області в діяльності команди та оцінювати ефективність політик безпеки.

Таким чином, модуль представлення результатів не тільки відображає інформацію, але й виступає визначальним фактором для прийняття рішень. Це дозволяє адміністратору відстежувати тенденції, встановлювати пріоритети реагування та приймати стратегічні рішення.

Підсумовуючи, концептуальна модель системи моніторингу чатів Telegram – це багаторівнева структура, що поєднує інтегровану систему управління з

платформою Telegram, семантичний та поведінковий аналіз, прийняття рішень, зберігання даних та зручну презентацію. Ця модель забезпечує комплексний контроль над комунікаційними потоками в організації, дозволяє виявляти приховані загрози, аналізувати дії користувачів та підтримувати загальний рівень інформаційної безпеки на належному рівні.

Головною перевагою концептуальної моделі є її гнучкість. Кожен модуль можна модернізувати або замінити залежно від технологічних розробок, появи нових алгоритмів або змін у корпоративній структурі. Завдяки такій архітектурі система здатна адаптуватися до нових загроз, забезпечуючи актуальність та ефективність підходів до захисту корпоративних чатів у Telegram.

3.4 Реалізація системи зберігання даних та організація інформаційних потоків

Після визначення концептуальної моделі наступним кроком є побудова детальної архітектури системи. Концептуальна модель визначає логічну структуру та загальні функціональні взаємозв'язки між модулями, тоді як архітектура розглядає ці модулі на більш технічному рівні, описуючи їх внутрішню організацію, методи взаємодії, протоколи обміну даними та механізми роботи.

Архітектура системи повинна відповідати вимогам масштабованості, надійності та безпеки, які є критично важливими для корпоративного середовища. Вона складається з декількох основних компонентів: серверної частини, що обробляє події; аналітичних модулів; бази даних; адміністративної панелі; допоміжних служб, що забезпечують логування, аудит та моніторинг самої системи. Важливо, щоб усі ці компоненти були не лише формально описані, а й узгоджені між собою за навантаженням, пропускною здатністю та вимогами до затримок, оскільки навіть добре спроєктований окремий модуль не зможе ефективно працювати у відриві від загальної архітектурної логіки. Саме тому на

етапі проектування значна увага приділяється не тільки функціям кожного елемента, а й сценаріям їхньої взаємодії в умовах реального використання.

Ключовою особливістю архітектури є її модульність: кожен компонент виконує окрему функцію, і зміна одного модуля не впливає на роботу іншого. Такий підхід спрощує впровадження оновлень, дозволяє системі масштабуватися горизонтально та швидко адаптуватися до зростання кількості користувачів. Модульність також дозволяє використовувати різні технології для реалізації різних частин системи, що підвищує її гнучкість та довгострокову життєздатність.

Важливою особливістю архітектури є використання асинхронної обробки подій. Оскільки повідомлення в корпоративних чатах можуть з'являтися з високою частотою, система повинна бути здатна обробляти їх одночасно, не блокуючи інші процеси. Це досягається за допомогою використання черг подій, асинхронних фреймворків та механізмів паралелізації.

База даних в архітектурі служить центральним сховищем інформації. Вона зберігає дані про повідомлення, користувачів, інциденти та журнали. Доступ до бази даних повинен бути швидким і надійним, тому важливо оптимізувати структури таблиць, індексацію та використання механізмів кешування.

Система моніторингу також повинна включати інтерфейс для адміністратора. Цей інтерфейс дозволяє переглядати інциденти, аналізувати статистику, керувати політиками безпеки та виконувати інші дії, пов'язані з управлінням системою. Інтерфейс повинен бути інтуїтивно зрозумілим і зручним, забезпечуючи можливість швидко реагувати на інциденти.

Таким чином, архітектура програмної системи є технічним втіленням концептуальної моделі і забезпечує її реалізацію в реальних умовах.

Щоб система моніторингу інформаційної безпеки корпоративних чатів у Telegram була не тільки теоретично правильною, але й практично придатною для реалізації, необхідно деталізувати архітектуру програмного забезпечення, тобто описати, як окремі компоненти реалізовані в програмному коді, як вони взаємодіють між собою і через які інтерфейси відбувається обмін даними. Архітектура програмної системи є своєрідним "скелетом" всього рішення, на який

вже накладаються конкретні алгоритми, реалізація логіки моніторингу та механізми зберігання і обробки інформації.

Як зазначалося вище, ключовою особливістю архітектури є її модульність. Це означає, що система складається з низки логічно відокремлених програмних компонентів, кожен з яких відповідає за певну частину загальної функціональності: один модуль перехоплює події з Telegram, інший відповідає за аналіз контенту, третій - за обробку інцидентів, а четвертий - за зберігання даних і формування звітів. Такий підхід дозволяє уникнути жорстких залежностей між окремими частинами коду, що, в свою чергу, значно спрощує обслуговування та розвиток системи.

З практичної точки зору реалізації, серверна частина системи може бути розгорнута як набір сервісів, що працюють у контейнерах. Наприклад, один контейнер містить модуль Telegram-боту, який отримує всі вхідні події; інший контейнер реалізує аналітичну частину, де обробляються текст, файли та поведінкові характеристики; третій контейнер працює як інтерфейс для доступу до бази даних; ще один контейнер може бути зарезервований для адміністративної панелі або веб-інтерфейсу. Завдяки такому розділенню кожен компонент можна масштабувати окремо, з урахуванням фактичного навантаження на нього.

Серверна частина боту реалізована як проєкт Node.js, структура якого визначається конфігураційним файлом package.json. Він задає основні параметри застосунку (назва, версія, точка входу), скрипти запуску сервісів та список зовнішніх бібліотек, необхідних для роботи системи. Зокрема, до залежностей входять пакети для взаємодії з Telegram Bot API, обробки аудіо- та відеофайлів, а також інтеграції з сервісами розпізнавання мови та зовнішніми моделями штучного інтелекту. Така конфігурація спрощує розгортання рішення на різних серверах і забезпечує централізоване керування версіями бібліотек.

Для забезпечення відтворюваності середовища виконання додатково використовується файл package-lock.json, в якому записуються точні версії всіх встановлених залежностей.

Файл опису конфігурації та залежностей боту показано на рис. 3.1.

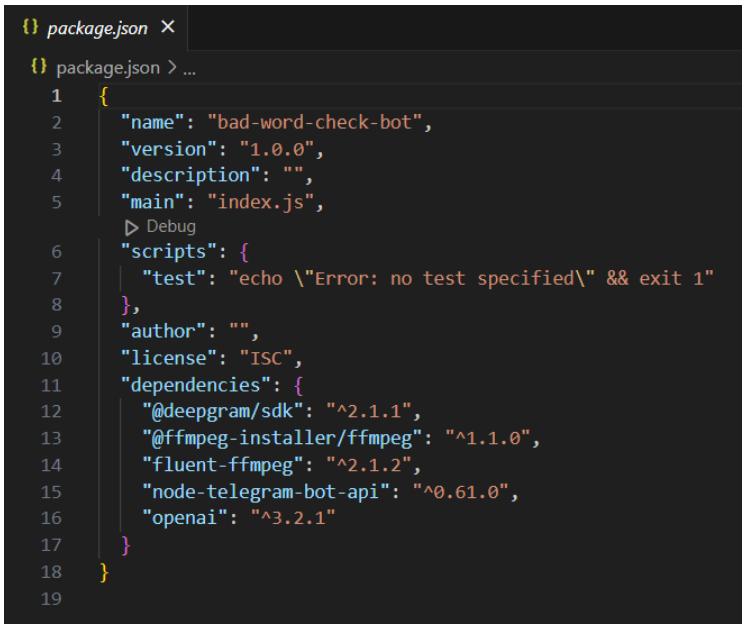


Рис. 3.1 Файл package.json з описом конфігурації та залежностей боту Telegram

При розгортанні системи в новому середовищі пакети встановлюються на основі цього файлу, що дозволяє уникнути несумісності, пов'язаної з прихованими оновленнями бібліотек, і забезпечує однакову конфігурацію програмного середовища для всіх установок системи.

Файл опису конфігурації та залежностей боту показаний на рис. 3.2.

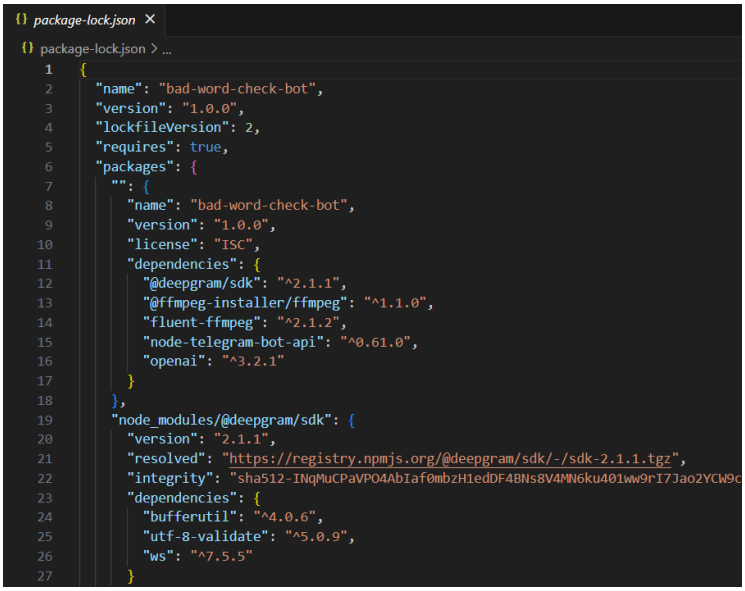


Рис. 3.2 Файл package.json з описом конфігурації та залежностей боту Telegram

Надзвичайно важливим елементом є архітектурна організація обміну повідомленнями між сервісами. У найпростішому випадку взаємодія може здійснюватися за допомогою прямих HTTP-запитів між модулями. Однак із збільшенням навантаження та зростанням складності системи часто доцільно використовувати проміжні черги повідомлень, які забезпечують буферизацію, асинхронність та підвищену стійкість до збоїв. У цьому випадку модуль, який отримує події з Telegram, не обробляє їх «на місці», а ставить у чергу, звідки їх забирають аналітичні працівники, обробляють і формують результати.

Іншим аспектом, який необхідно враховувати при проектуванні архітектури програмного забезпечення, є розподіл обов'язків всередині модулів. Наприклад, сам аналітичний модуль може включати кілька підкомпонентів: окремий підкомпонент для аналізу сигнатур тексту, окремий для застосування моделей машинного навчання, а також інший для поведінкового аналізу та агрегації статистики за певні проміжки часу. Такий внутрішній розподіл дозволяє змінювати або вдосконалювати окремі підкомпоненти без необхідності повного перепроектування всього модуля.

Значну увагу також слід приділити механізмам обробки помилок. Будь-яка реальна система, особливо та, що працює в мережевому середовищі, повинна бути здатна витримувати тимчасові збої, помилки підключення до API Telegram, несправності бази даних або перевантаження сервера. Архітектура повинна включати механізми повторних спроб, реєстрацію помилок і повідомлення адміністратора в разі критичних збоїв. Важливо, щоб локальний збій в одному модулі не призводив до вимкнення всієї системи: наприклад, тимчасова втрата доступу до бази даних не повинна блокувати прийом нових подій – в цьому випадку дані можуть тимчасово накопичуватися в черзі або буфері.

Безпека архітектури системи також є пріоритетом. Усі зовнішні точки входу, через які Telegram надсилає оновлення, повинні бути захищені. Використання шифрування, перевірка підписів запитів та обмеження доступу за IP-адресами утворюють додатковий рівень захисту. Не менш важливим є розділення прав

доступу між модулями: не всі компоненти повинні мати повний доступ до всієї бази даних, а лише до тих таблиць і полів, які необхідні для їхньої роботи.

Особливу роль відіграє модуль адміністрування, який виступає «обличчям» системи для відповідальних осіб в організації. З архітектурної точки зору він може бути реалізований як веб-додаток або через адміністративний інтерфейс у самому Telegram, де адміністратор взаємодіє з окремим ботом, який надає йому інформацію в зручній формі. У будь-якому випадку цей модуль повинен мати доступ до бази даних, отримувати інформацію про інциденти, генерувати зведені звіти та, за необхідності, змінювати налаштування політики безпеки.

В кінцевому підсумку, програмна архітектура системи моніторингу є результатом поєднання вимог до продуктивності, безпеки, модульності та простоти використання. Це не статична структура, яка раз і назавжди, а скоріше гнучка структура, що дозволяє подальший розвиток: заміну окремих компонентів, оновлення алгоритмів, інтеграцію нових аналітичних модулів або додаткових каналів моніторингу. Саме це дозволяє системі залишатися актуальною в умовах постійно мінливого інформаційного середовища та еволюції загроз.

3.5 Методика проведення експереминтального дослідження роботи системи

Алгоритмічний компонент системи моніторингу інформаційної безпеки корпоративного чату Telegram – це рівень, на якому абстрактні вимоги, концептуальні моделі та архітектурні рішення перетворюються на конкретні механізми виявлення інцидентів. Якщо в попередніх розділах йшлося про те, що повинна робити система і як вона побудована, то на рівні алгоритмів дається відповідь на питання, як аналізуються події і як система робить висновок про потенційну небезпеку того чи іншого повідомлення, файлу або послідовності дій користувача.

Серед усіх елементів програмної реалізації саме алгоритмічний блок найбільше визначає ефективність системи: його якість визначає точність і повноту виявлення порушень, кількість помилкових спрацьовувань, здатність працювати під змінним навантаженням і адаптуватися до нових типів загроз. Алгоритми повинні враховувати, що корпоративна комунікація є живим, динамічним процесом, де зміст повідомлень часто не є чітко формалізованим, а деякі ризиковані дії маскуються під звичайну робочу діяльність. Тому простих перевірок на наявність заборонених слів тут явно недостатньо.

В рамках системи моніторингу було розроблено та впроваджено кілька взаємопов'язаних груп алгоритмів: алгоритми аналізу текстового контенту, алгоритми обробки вкладень і файлів, алгоритми аналізу поведінки користувачів та алгоритми контекстуально-семантичної інтерпретації. Кожна з цих груп відіграє свою роль, але всі вони працюють разом, доповнюючи одна одну і формуючи узагальнену картину ризику для конкретного повідомлення або події.

Текстові повідомлення є основним носієм інформації в корпоративних чатах Telegram, і саме через них найчастіше відбуваються як навмисні, так і ненавмисні порушення політики безпеки. Алгоритми аналізу тексту базуються на поєднанні класичних підходів на основі сигнатур, статистичних методів та моделей обробки природної мови. На рівні реалізації це означає, що кожне повідомлення, яке надходить до системи, проходить кілька послідовних етапів обробки.

Спочатку виконується попередня нормалізація тексту. Її мета – привести повідомлення до форми, яку можна порівняти з еталонними шаблонами та передати для семантичного аналізу. На цьому етапі система очищає текст від непотрібних службових символів, уніфікує регістр і видаляє типові елементи «шуму», що не несуть семантичного значення. Одночасно виділяються окремі лексеми – слова, числові послідовності та символи, які можуть бути важливими (наприклад, номери рахунків, ідентифікаційні коди, посилання на конкретні системи тощо).

Після нормалізації текст порівнюється з базою даних сигнатур, яка містить заборонені слова, фрази та вирази, характерні для порушень політики, посилання

на конфіденційні дані, а також типові фрази, пов'язані із соціальною інженерією. Тут використовується не тільки пряме порівняння, але й механізми часткового збігу для виявлення ситуацій, коли користувачі навмисно змінюють написання слів, щоб обійти контроль системи. Однак цього рівня аналізу недостатньо, оскільки одне й те саме слово в різних контекстах може мати абсолютно різні наслідки для безпеки.

На практичному рівні словник сигнатур реалізований у вигляді окремого файлу disallowedWordlist.json, який містить ключові слова та вирази, що не допускаються в діловому спілкуванні. Він включає, зокрема, слова, пов'язані з насильством, війною, нецензурною лексикою або іншими темами, що суперечать корпоративній політиці. Під час обробки кожного повідомлення система порівнює його вміст із цим списком і, якщо є збіг, підвищує базовий рівень ризику для відповідної події.

Фрагмент файлу зі словником заборонених слів зображений на рис. 3.3.

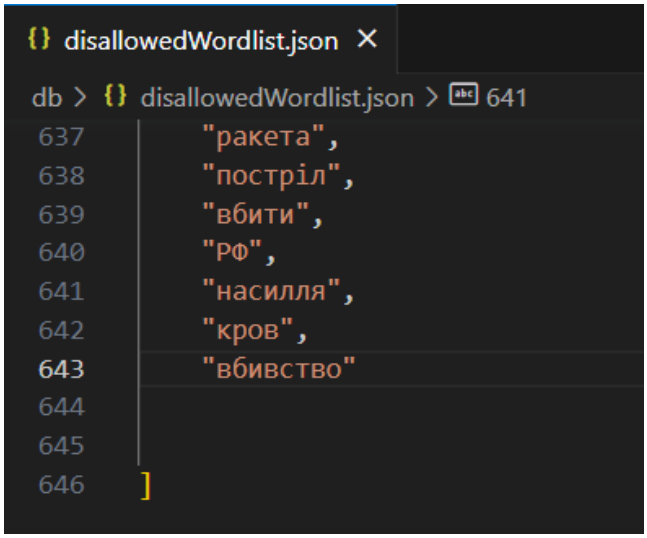


Рис. 3.3 Фрагмент файлу disallowedWordlist.json зі словником заборонених слів

Саме тому наступним кроком є контекстний аналіз. Тут система вже не обмежується перевіркою окремих слів, а розглядає повідомлення в цілому, визначаючи його семантику, тобто значення. Для цього використовуються моделі обробки природної мови, щоб оцінити, чи стосується повідомлення фактичної передачі конфіденційних даних, чи це лише теоретичне обговорення, навчальний

приклад або навіть жарт. Такі моделі допомагають уникнути ситуацій, коли кожне згадування слова «пароль» або «контракт» автоматично інтерпретується як інцидент.

Разом алгоритми аналізу текстового вмісту формують базову оцінку ризику для кожного повідомлення. Вона може бути низькою, якщо виявлено лише незначні збіги сигнатур, або може значно зрости, коли семантичний аналіз показує, що користувач фактично передає конфіденційні дані або порушує встановлені політики. Потім цей результат передається до модуля оцінки ризиків, де він враховується при прийнятті комплексного рішення.

Не менш важливим джерелом ризику є передача файлів через корпоративні чати. Документи, таблиці, презентації, архіви – все це потенційні носії конфіденційної інформації, яка не повинна виходити за межі внутрішнього інформаційного простору компанії. У зв'язку з цим система реалізує окрему групу алгоритмів, спрямованих на аналіз вкладень.

Кожен файл, що надсилається в чат, супроводжується мінімальною кількістю метаданих – назвою, розміром, типом, а також інформацією про відправника та час надсилання. Алгоритм аналізу починається з перевірки типу файлу та його відповідності дозволеним форматам. Деякі типи можуть вважатися безпечнішими, а інші – більш ризикованими. Наприклад, архіви можуть приховувати кілька документів, у тому числі небажані, а виконувані файли або складні формати можуть містити шкідливий код.

Назва файлу також дуже важлива. Вона часто містить непрямі посилання на характер його вмісту: згадки про імена клієнтів, внутрішні проекти, фінансові звіти, номери контрактів, конфіденційні документи. Алгоритм аналізу назви файлу порівнює її зі списком ключових слів і шаблонів, визначених у політиках безпеки. Це дозволяє виявляти навіть ті випадки, коли система не виконує глибокий аналіз вмісту з міркувань продуктивності, а замість цього покладається на назву та тип файлу як індикатори ризику.

Якщо виявлено підвищену підозру, може бути застосований розширений аналіз. Він передбачає читання частини вмісту, пошук ознак конфіденційних даних

та використання методів розпізнавання тексту, вбудованих у PDF-файли або зображення. Цей етап є більш ресурсоємним, тому зазвичай застосовується тільки до файлів з високим пріоритетом або в умовах помірного навантаження.

Алгоритми аналізу вкладень також враховують поведінковий контекст. Наприклад, якщо один і той самий користувач надсилає велику кількість великих файлів за короткий проміжок часу, це може свідчити про масове скидання даних. У такій ситуації система враховує загальний обсяг переданих файлів, їхні імена, час надсилання та роль відправника в організації. Ці фактори створюють підвищений рівень ризику, навіть якщо окремий файл не виглядає підозрілим.

Механізми поведінкового аналізу складають окрему групу алгоритмів у системі. Якщо аналіз тексту та файлів зосереджується на змісті повідомлень, то поведінковий аналіз звертає увагу на динаміку дій користувачів – їхню активність у часі, характер взаємодії з іншими учасниками та нетипові зміни в стилі спілкування.

На практиці це означає, що система створює умовний «профіль поведінки» для кожного користувача, який накопичує інформацію про середню кількість повідомлень на день, типовий час активності, частоту передачі файлів, схильність до використання певних тем тощо. Важливо не тільки фіксувати ці параметри, але й відстежувати їх зміни. Раптове збільшення активності, особливо в незвичайний час, може свідчити про те, що обліковий запис було зламано або що користувач розпочав діяльність, пов'язану з витоком даних.

Аналіз поведінки також включає виявлення аномалій. Не кожне відхилення є порушенням, але значні або систематичні аномалії представляють інтерес для служб безпеки. Якщо, наприклад, співробітник, який зазвичай надсилає кілька коротких робочих повідомлень на день, раптом починає активно завантажувати файли вночі, система розглядає це як подію, що вимагає підвищеної уваги. Аналогічно, незвично агресивні або емоційні моделі спілкування можуть сигналізувати про внутрішній конфлікт, який зрештою переросте в навмисні порушення політики.

Алгоритми поведінкового аналізу не працюють ізольовано: їхні результати інтегруються з даними аналізу тексту та файлів. Це дозволяє системі генерувати більш реалістичні оцінки ризиків, враховуючи не тільки зміст конкретного повідомлення, але й те, як воно вписується в загальну поведінкову модель користувача.

Останній рівень алгоритмічного аналізу пов'язаний з контекстуально-семантичною інтерпретацією подій. Якщо попередні алгоритми зосереджувалися на окремих вимірах – текстовому вмісті, характеристиках файлів, поведінці користувача – на цьому етапі вони інтегруються в єдину картину. Мета контекстуально-семантичного аналізу полягає не просто в тому, щоб знайти «ризиковані» елементи, а в тому, щоб зрозуміти їх місце в процесі комунікації.

Система намагається відповісти на такі питання: чи є конкретне повідомлення або послідовність дій логічним продовженням попередніх дискусій, чи порушує воно нормальний контекст? Чи відповідає зміст повідомлення ролям, які відправник має в реальній організаційній структурі? Чи може поєднання кількох, на перший погляд, нейтральних подій становити загрозу? Такий підхід дозволяє виявляти складні сценарії, в яких жоден окремий епізод не виглядає підозрілим, але низка дій у сукупності сигналізує про витік інформації або підготовку до нього.

Для реалізації таких алгоритмів використовуються як класичні методи аналізу послідовностей, так і сучасні моделі нейронних мереж, здатні враховувати контекст на рівні діалогу. Система аналізує не тільки окремі фрази, але й цілі ланцюжки питань і відповідей, переходи між темами та повторення певних мотивів. Це дозволяє відрізнити нормальну робочу діяльність від навмисних спроб обійти політику безпеки.

Врешті-решт, результати контекстуально-семантичного аналізу передаються до модуля прийняття рішень, де вони поєднуються з оцінками, отриманими за допомогою інших алгоритмів. Така інтеграція дозволяє провести найбільш повну та обґрунтовану оцінку ризиків, а отже, прийняти більш обґрунтовані рішення

щодо повідомлення адміністратора, блокування повідомлень або подальшого моніторингу поведінки користувачів.

Узагальнена послідовність обробки подій у розробленій системі продемонстрована алгоритмом боту Telegram для моніторингу інформаційної безпеки. На схемі показані етапи ініціалізації, моніторингу сервісного чату, аналізу змісту повідомлень, прийняття рішень про наявність порушень політики та виконання відповідних реакцій.

Алгоритм роботи Telegram-боту зображений на рис. 3.4.

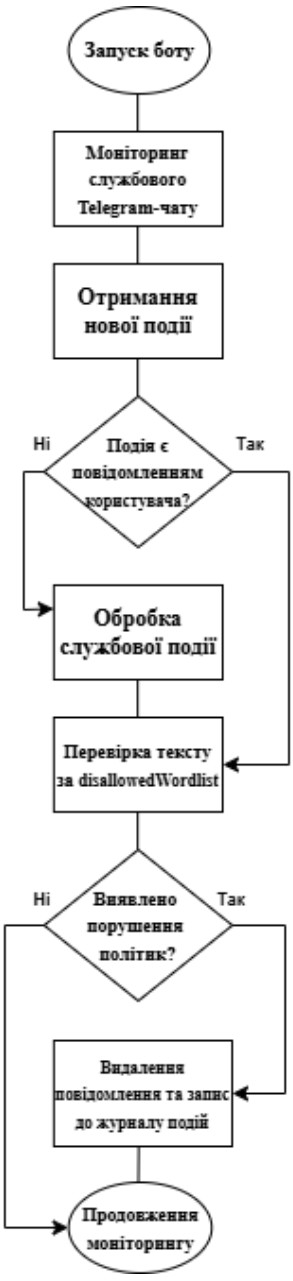


Рис. 3.4 Алгоритм роботи боту Telegram для моніторингу інформаційної безпеки в чаті сервісу

3.6 Оцінка ефективності системи та аналіз результатів

Проектування структури бази даних та організація інформаційних потоків є одним з ключових етапів створення системи моніторингу інформаційної безпеки в корпоративних чатах Telegram, оскільки саме на цьому рівні визначається, як система буде зберігати, пов'язувати та обробляти дані, що надходять із робочого середовища. Успіх моделі даних визначає не тільки продуктивність системи, але й її аналітичні можливості, гнучкість у розширенні та надійність у довгостроковій експлуатації.

Система моніторингу працює з неоднорідною інформацією. Частина даних є структурованою, наприклад ідентифікатори користувачів, часові мітки, типи подій та рівні ризику. Решта є напівструктурованою або слабо формалізованою: тексти повідомлень, імена файлів та результати семантичного аналізу. До цього додаються журнали системних подій, політики безпеки, історія інцидентів та профілі поведінки. Тому модель бази даних повинна одночасно підтримувати чіткі логічні відносини між об'єктами, забезпечувати швидкий доступ до даних і дозволяти виконувати складні запити для подальшого аналізу.

Структура бази даних базується на ідеї відокремлення ключових об'єктів один від одного, але з підтримкою чітких посилань між ними. Таким чином, інформація про користувачів, повідомлення, файли, інциденти, політики безпеки, результати аналізу та системні журнали не зберігається в одному «монолітному» просторі, а розподіляється між окремими логічними таблицями. Це дозволяє уникнути дублювання даних, зменшує ризик помилок і забезпечує нормалізовану структуру, де кожна таблиця відповідає за свій аспект системи.

Одним із центральних елементів моделі є об'єкт, що описує користувачів корпоративного чату. Для кожного користувача записуються такі характеристики, як унікальний Telegram ID, ім'я для відображення, псевдонім, можлива роль у чаті (звичайний учасник, адміністратор, бот), час першої появи та останньої активності. Ці дані не тільки дозволяють відстежувати історію участі людини в комунікаціях, але й використовуються в модулі поведінкового аналізу. Відповідно, таблиця

користувачів стає опорою для багатьох інших сутностей – з нею пов'язані повідомлення, інциденти, поведінкові профілі та інші об'єкти.

Для зберігання інформації про учасників корпоративного чату реалізований прототип використовує окремий конфігураційний файл users.json, який фактично діє як проста нереляційна таблиця користувачів. Він зберігає ідентифікатор користувача Telegram, інформацію про те, чи є він ботом, його ім'я, псевдонім, статус адміністратора та поля сервісу, пов'язані з поведінковим аналізом для кожного облікового запису: кількість попереджень, кількість блокувань, дата приєднання до чату тощо. Така структура дозволяє системі швидко отримувати доступ до профілю користувача під час обробки кожного повідомлення та враховувати його історію при розрахунку рівня ризику.

Структуру файлу для зберігання інформації про користувачів чату зображено на рис. 3.5.

Рис. 3.5 Структура файлу users.json для зберігання інформації про користувачів чату Telegram

Не менш важливою є таблиця, відповідальна за повідомлення. Вона містить інформацію про зміст кожного повідомлення, час його відправлення, автора, тип (звичайний текст, сервіс, з вкладенням, відповідь або пересилання) та атрибути сервісу, такі як посилання на ідентифікатор джерела в Telegram. Текст повідомлення зберігається як окреме поле, але водночас може використовуватися для побудови індексів або зберігатися в додаткових структурах для повнотекстового пошуку. Для системи моніторингу важливо, щоб таблиця повідомлень стала «хабом», через який проходять більшість аналітичних операцій: від оцінки змісту до зв'язку з конкретними інцидентами.

Тісно пов'язаною з повідомленнями є суть, що описує вкладення. Кожен файл, надісланий у чаті, пов'язаний з конкретним повідомленням, але має свої власні атрибути: тип, розмір, оригінальна назва, контрольна сума, а також ознаки його можливої класифікації як конфіденційного. Наявність окремої таблиці файлів дозволяє проводити аналіз не тільки на рівні тексту, але й на рівні потоків передачі документів, виявляти користувачів з підозрілою активністю, складати статистику передачі різних типів файлів і реалізовувати механізми блокування або карантину файлів, які система оцінює як ризикові.

Інциденти займають особливе місце в структурі бази даних. Кожен інцидент зазвичай пов'язаний з одним або декількома повідомленнями і вказує на подію, яка була класифікована як порушення або потенційна загроза. Тут фіксуються рівень ризику, тип інциденту, порушена політика, час виявлення, короткий опис та статус обробки (новий, в процесі, закритий). Така структура дозволяє не тільки фіксувати порушення, але й відстежувати весь життєвий цикл інциденту – від моменту його виникнення до остаточного вирішення.

Для збереження гнучкості системи важливо мати окрему модель для опису політик безпеки. Вона зберігає формалізовані правила, які використовуються в алгоритмах моніторингу: список заборонених ключових слів і фраз, порогові параметри для поведінкового аналізу, дозволені типи файлів, умови для підвищення або зниження рівня ризику. Це дозволяє змінювати політики, не

втручаючись у код самої системи: адміністратор може оновлювати записи в базі даних, і нові правила будуть автоматично враховуватися при обробці подій.

Іншим необхідним компонентом є таблиці, що описують результати аналітичної обробки. Наприклад, для кожного повідомлення можуть зберігатися числові або категоріальні оцінки, отримані з модулів аналізу тексту, пошуку сигнатур, поведінкових алгоритмів і моделей NLP. Така деталізація дозволяє проводити більш глибокий аналіз роботи системи, оцінку якості алгоритмів, перебудову моделей і створення складних звітів , що показують внесок кожного аналітичного модуля в остаточну оцінку ризику.

Окремої згадки заслуговують системні журнали. Для надійної роботи системи необхідно, щоб усі технічні події, помилки, перезапуски модулів, зміни конфігурації та дії адміністратора записувалися в окремі таблиці. Такі журнали дозволяють проводити внутрішні аудити, відновлювати хід подій у разі збоїв або підозрілої активності, а також оцінювати стабільність системи. Це додає прозорості, підвищує керованість рішення та дозволяє виявляти не тільки загрози від користувачів, але й можливі проблеми в самій інфраструктурі.

Не менш важливим, ніж логічна структура, є питання організації інформаційних потоків. Вони визначають, як дані переміщуються між різними частинами системи: від точок входу до моменту потрапляння в базу даних і подальшого використання в аналітиці або звітності. У спрощеному вигляді основний шлях виглядає так: подія в корпоративному чаті – її перехоплення через Telegram Bot API – передача до модуля збору – попередня обробка та нормалізація – аналітична обробка – збереження результатів та формування інцидентів – інформування адміністратора.

На практиці кожен з цих етапів може бути реалізований як окремий програмний компонент, але логіка залишається незмінною: дані повинні переміщатися послідовно, без втрат, з чіткою фіксацією кожної операції. Важливо забезпечити таку структуру інформаційних потоків, щоб у разі тимчасового збою одного з модулів система могла відновити роботу і обробити накопичені події. Для

цього можуть використовуватися проміжні черги, буфери або механізми транзакційної реєстрації, що гарантують цілісність даних.

При проектуванні інформаційних потоків також необхідно враховувати розподіл навантаження. Наприклад, модуль, який отримує оновлення з Telegram, не повинен виконувати важкі аналітичні операції – його завдання полягає в тому, щоб якомога швидше записати подію і передати її на наступний рівень обробки. Аналітичні модулі, в свою чергу, можуть працювати асинхронно, використовуючи дані, записані в чергах або проміжних таблицях. Такий підхід забезпечує стійкість системи до пікових навантажень: навіть якщо в певний момент надходить надмірна кількість повідомлень, вони не блокують роботу всієї системи, а обробляються по мірі появи вільних ресурсів.

Важливим є також питання структурування історичних даних. У міру накопичення інформації база даних може значно зростати в розмірах. Щоб уникнути падіння продуктивності, доцільно використовувати механізми архівування старих записів, поділу таблиць за часовими інтервалами та використання додаткового сховища для даних, що рідко запитуються. При цьому структура інформаційних потоків повинна забезпечувати введення нових записів в активні таблиці, а старі поступово переміщувати в архіви, звідки їх все ще можна отримати для аналітики, але вони не впливають на операційну роботу системи.

Підсумовуючи, можна сказати, що проектування структури бази даних та інформаційних потоків у системі моніторингу корпоративного чату Telegram спрямовано на досягнення балансу між продуктивністю, гнучкістю та надійністю. Добре спроектована модель даних забезпечує ефективний аналіз, зручне адміністрування та масштабування системи у міру зростання обсягів комунікації. Правильно організовані інформаційні потоки гарантують, що жодна важлива подія не буде втрачена, а всі інциденти будуть записані, проаналізовані та представлені адміністраторам у формі, придатній для прийняття рішень.

Висновки до розділу 3

У третьому розділі наведено комплексний проект та опис реалізації системи моніторингу інформаційної безпеки корпоративних чатів у Telegram на основі теоретичних принципів, сформульованих у попередніх розділах. Показано перехід від концептуальних моделей до конкретних інженерних рішень, з урахуванням вимог модульності, масштабованості, відмовостійкості та мінімального втручання в звичні комунікаційні процеси користувачів. Обґрунтовано, що система має працювати у фоновому режимі, забезпечуючи безперервний контроль за потенційно небезпечними діями без зміни характеру повсякденного спілкування.

На основі концептуальної моделі розроблено архітектуру програмної системи як набору взаємопов’язаних, але відносно незалежних модулів, які реалізують інтеграцію з Telegram, збір і попередню обробку подій, аналітику, оцінку ризиків, прийняття рішень і зберігання даних. Описано механізми асинхронної обробки, можливість використання черг повідомлень, підходи до розподілу навантаження та масштабування. Важливим результатом стала розробка алгоритмічної бази системи: алгоритмів аналізу текстових повідомлень, вкладень та поведінкової активності користувачів, які у сукупності формують багатовимірну оцінку загрози для кожної події.

Окремо спроектовано структуру бази даних та інформаційних потоків, визначено ключові сутності (користувачі, повідомлення, файли, інциденти, політики, журнали, аналітичні результати) та логіку їх взаємозв’язків. Показано, що така модель забезпечує можливість реконструкції подій, аналізу сценаріїв поведінки та формування звітів, а також підтримує масштабування та історичне зберігання великих обсягів інформації. У підсумку третій розділ формує цілісне бачення системи як програмно-технічного рішення, у якому теоретичні напрацювання попередніх розділів набувають конкретної реалізації у вигляді архітектури, алгоритмів і структур даних, придатних до практичного впровадження в корпоративному середовищі.

ВИСНОВКИ

У магістерській роботі досягнуто поставленої мети – розроблено систему моніторингу інформаційної безпеки корпоративних чатів у месенджері Telegram, яка забезпечує контроль дотримання політик безпеки, виявлення порушень та формування інцидентів із можливістю сповіщення адміністратора. На основі аналізу предметної області обґрунтовано доцільність використання Telegram у корпоративній комунікації та одночасно виявлено обмеження платформи з погляду інформаційної безпеки. Це дало змогу сформувати вимоги до системи моніторингу й чітко визначити коло завдань дослідження.

У межах поставлених завдань виконано аналіз корпоративної комунікації в месенджерах, класифіковано загрози (технічні, організаційні та пов'язані з людським фактором), розглянуто наявні підходи та засоби контролю (модераційні боти, DLP, SIEM/SOAR, спеціалізовані рішення) та показано їхню обмежені придатність саме для корпоративних чатів Telegram. На цій основі розроблено концепцію системи моніторингу, побудовано концептуальну модель та архітектуру програмного рішення, спроектовано структуру бази даних і інформаційних потоків, а також сформовано алгоритми аналізу тексту, вкладень і поведінки користувачів. Завершальним етапом стало створення прототипу на базі Telegram-боту, реалізація основних модулів (збір подій, аналіз, реєстрація інцидентів, сповіщення адміністратора) та експериментальна перевірка на тестових сценаріях, що підтвердила здатність системи виявляти типові порушення політик і зменшувати потребу в ручній модерації. Отримані результати можуть бути використані для впровадження системи моніторингу в реальних корпоративних середовищах, а також як основа для подальшого розширення функціоналу.

Наукова новизна одержаних результатів полягає у розробці спеціалізованої моделі та архітектури системи моніторингу інформаційної безпеки корпоративних чатів у месенджері Telegram, орієнтованої на використання месенджера як корпоративного каналу зв'язку. Запропонований підхід поєднує сигнатурні методи контролю з поведінковим та контекстно-семантичним аналізом подій, а також

формалізує вимоги, структуру даних і логіку взаємодії компонентів системи для автоматичного виявлення порушень політик безпеки та подальшої інтеграції рішення в існуючу інфраструктуру підприємства.

Подальший розвиток результатів цієї роботи може включати в себе інтеграцію методів штучного інтелекту та машинного навчання в усі етапи моніторингу корпоративних чатів. Перспективними є адаптивні моделі поведінкової аналітики, що самонавчаються і здатні виявляти інсайдерські загрози, а також застосування сучасних мовних моделей для точнішого аналізу повідомлень і виявлення шахрайських повідомлень. Важливими напрямами є розширення системи на інші месенджери та канали комунікації, інтеграція з DLP-, SIEM- та SOAR-рішеннями, а також удосконалення інтерфейсів адміністратора з використанням підходів ШІ, що дозволить не лише фіксувати інциденти, а й зрозуміло пояснювати причини їх виявлення.

ПЕРЕЛІК ПОСИЛАНЬ

1. Горовий В.М. *ІТ-субкультура як фактор розвитку сучасного правотворення,* 2012.
2. Alireza Bahramali, Ramin Soltani, Amir Houmansadr, Dennis Goeckel, Don Towsley. *Practical Traffic Analysis Attacks on Secure Messaging Applications*. Режим доступу: https://arxiv.org/abs/2005.00508Buyya, R., Broberg, J., Goscinski, A. *Cloud Computing: Principles and Paradigms*. Wiley, 2011.
3. Stallings W. *Network Security Essentials: Applications and Standards.* 6th ed. – Pearson, 2017.
4. Unger N. et al. *SoK: Secure Messaging*, 2015. Режим доступу: https://www.ieee-security.org/TC/SP2015/papers-archived/6949a232.pdf.
5. Lee J. et al. *Security Analysis of End-to-End Encryption in Telegram*, 2017. Режим доступу: https://caislab.kaist.ac.kr/publication/paper_files/2017/SCIS17_JU.pdf.
6. Горовий В. М. *ІТ-субкультура як фактор розвитку сучасного правотворення* / В. М. Горовий *Актуальні проблеми управління інформаційною безпекою держави*: мат. наук-практ. конф.: Наук-вид. відділ НА СБ України, 2012.
7. Simpson: *“ You Don't Know JS”*, 2015.
8. David Flanagan: *"JavaScript: The Definitive Guide"*, 2011.
9. Смірнова Т. *Дослідження сучасного стану SIEM-систем. Центральноукраїнський науковий вісник (інформаційна безпека),* 2024. Режим доступу: https://csecurity.kubg.edu.ua/.
10. Microsoft. *What is Data Loss Prevention (DLP)? – Microsoft Security Documentation.* Режим доступу: https://www.microsoft.com/en-us/security/business/security-101/what-is-data-loss-prevention-dlp.
11. ESET. *Telegram Privacy Explained: What’s Protected & What’s Not. ESET Security Blog*, 2025. Режим доступу: https://www.eset.com/.
12. Stoyan Stefanov: "*JavaScript Patterns*", 2010. Режим доступу до ресурсу: https://sd.blackball.lv/library/JavaScript_Patterns_%282010%29.pdf.

13. NordStellar. *Telegram Monitoring: How to Prevent Cyber Risks and Detect Leaked Data*, 2025. Режим доступу: https://nordstellar.com/blog/telegram-monitoring/.
14. Kumar A. *Anomaly Detection in Chatbot Interactions for Security Enhancement*, 2025. Режим доступу: https://www.sciencedirect.com/science/article/abs/pii/S0957417425023875.
15. Tuvis. *What Is Data Loss Prevention in Messaging Apps*, 2025. Режим доступу: https://tuvis.com/data-loss-prevention-messaging-apps/.
16. MailSPEC. *A Guide to Data Loss Prevention (DLP) for Enterprise Communications*, 2024. Режим доступу: https://www.mailspec.com/post/a-guide-to-data-loss-prevention-dlp-for-enterprise-communications.
17. Unger N., Dechand S., Bonneau J. *SoK: Secure Messaging // IEEE Symposium on Security and Privacy*, 2015. Режим доступу: https://cacr.uwaterloo.ca/techreports/2015/cacr2015-02.pdf.
18. Miculan M., Vitacolonna N. *Automated Symbolic Verification of Telegram's MTProto 2.0 // Proceedings of the 10th International Conference on Security for Information Technology and Communications*, 2021. Режим доступу: https://www.scitepress.org/Papers/2021/105496/105496.pdf.
19. Saribekyan H., Margvelashvili A. *Security Analysis of Telegram // Computer and Network Security, Final Project*, 2017. Режим доступу: https://courses.csail.mit.edu/6.857/2017/project/19.pdf.
20. Lee J., Kwon T., Kim J. *Security Analysis of End-to-End Encryption in Telegram // Symposium on Cryptography and Information Security*, 2017. Режим доступу: https://caislab.kaist.ac.kr/publication/paper_files/2017/SCIS17_JU.pdf.
21. Lee J., Kwon T., Kim J. Security Analysis of End-to-End Encryption in Telegram // Symposium on Cryptography and Information Security, 2017. Режим доступу: https://caislab.kaist.ac.kr/publication/paper_files/2017/SCIS17_JU.pdf.
22. National Cyber Security Centre (NCSC, UK). *Choosing an Enterprise Instant Messaging Solution*, 2020. Режим доступу: https://www.ncsc.gov.uk/collection/device-security-guidance/policies-and-settings/choosing-an-enterprise-instant-messaging-solution.

23. CISA. *Mobile Communications Best Practice Guidance. Cybersecurity and Infrastructure Security Agency*, 2024. Режим доступу: https://www.cisa.gov/sites/default/files/2024-12/guidance-mobile-communications-best-practices.pdf.
24. Amazon Web Services (AWS). *5 Keys to Secure Enterprise Messaging*, 2023. Режим доступу: https://d1.awsstatic.com/Security/eBook_5_Keys_to_Secure_Enterprise_Messaging_2023.pdf.
25. LeapXpert. *5 Tips on Enterprise Messaging Security for Better Communication*, 2022. Режим доступу: https://www.leapxpert.com/5-tips-on-enterprise-messaging-security-for-better-communication/.
26. *The danger of using Telegram and its impact on*, CSecurity – Науковий журнал з кібербезпеки, 2024. Режим доступу: https://csecurity.kubg.edu.ua/index.php/journal/article/view/648.
27. Rittinghouse J., Ransome J. *IM Instant Messaging Security*, 2005. Режим доступу: https://shop.elsevier.com/books/im-instant-messaging-security/rittinghouse-phd-cism/978-1-55558-338-5.
28. Miculan M., Vitacolonna N. *Automated Symbolic Verification of Telegram's MTProto 2.0*, 2020. Режим доступу: https://arxiv.org/abs/2012.03141.
29. Abid N. *Advancements and Best Practices in Data Loss Prevention: A Comprehensive Review*, 2024. Режим доступу: https://www.researchgate.net/publication/387325365_Advancements_and_Best_Practices_in_Data_Loss_Prevention_A_Comprehensive_Review.
30. Lymishchenko V. *Telegram and Dark Web Platforms in Digital Criminal Ecosystems*, 2025. Режим доступу: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=5722747.

ВИХІДНИЙ КОД ОСНОВНОГО МУДУЛЮ TELEGRAM-БОТУ

Файл index.js

```
const TelegramBot = require('node-telegram-bot-api');
const fs = require('fs');
const token = '6010781888:AAFleyZDGJsXnvMTZ3GuMQAIxs2pRtxPPD0';
const transcribe = require('./controllers/deepgramTranscription');
const converter = require('./controllers/oggToMP3');
const audioDirectory = './audio/';
const isBadWord = require('./controllers/isBadWord');
const disallowedWordlist = require('./db/disallowedWordlist.json');
const bot = new TelegramBot(token, {
    polling: true
});
const groupID = -1001974583370;
const users = JSON.parse(fs.readFileSync('./db/users.json', 'utf-8', (error) => { if
(error) throw Error(error) })) || [];
function user(chatID) { return users.filter(user => user.id === chatID)[0]; }
function addNewUser(info, chat) {
    let object = { ...info, canSayBadWords: false, isAdmin: false, warned: false,
timesWasBanned: 1, joinedAt: new Date().getTime() };
    if (chat.id !== groupID) return;
    users.push(object);
    updateUsers();
    return;
}
function updateUsers() {
    fs.writeFileSync('./db/users.json', JSON.stringify(users), 'utf-8', (error) => {
        if (error) console.error(error);
    });
}
bot.on('left_chat_member', (msg) => {
    const { left_chat_member } = msg;
    const leftUserIndex = users.findIndex(_user => _user.id === left_chat_member.id);
    users.splice(leftUserIndex, 1);
    updateUsers();
})
bot.on('message', async (msg) => {
    const { voice, chat, from } = msg;
    const fromID = from.id;
    const chatID = chat.id;
    if (chatID !== groupID) return;
    if (!user(fromID)) addNewUser(from, chat);

    if (voice) {
        console.log('voice detected');
        const { file_id } = voice;
```

```
        bot.downloadFile(file_id, audioDirectory).then(savedFilePath => {
            converter(`${__dirname}\\${savedFilePath}`,
`${__dirname}\\${savedFilePath.replace(/(\.oga|\.ogg)/g, '.mp3')}`)
                .then(path => {
                    transcribe(path).then(data => {
                        if (!user(fromID).isAdmin &&
isBadWord(disallowedWordlist.join(','), data.transcript)) {
                            bot.deleteMessage(chatID, msg.message_id);

                            if (!user(fromID).warned) {
                                bot.sendMessage(chatID,
`⚠<b>Попередження!</b>⚠\n<i>В цьому чаті не можна вживати лайливі висловлювання!
Наступного разу вас буде заблоковано на 5хв, кожного наступного разу термін
блокування буде зростати.</i>`, { parse_mode: 'HTML' });
                                user(fromID).warned = true;
                                updateUsers();
                                return;
                            }
                            const period = new Date().getTime() + (10000 *
user(fromID).timesWasBanned);

                            bot.restrictChatMember(chatID, fromID, { until_date:
period }).then(response => {
                                if (response) {
                                    bot.sendMessage(chatID, `🚫<b>Юзера
${from.first_name || from.last_name} було заблоковано</b>🚫\n<b>Причина</b>:
<i>вживання заборонених слів</i>\n<b>Дата відновлення користуванням чату</b>:
<i>${new Date(period).toLocaleString('uk-UA', { timeZone: 'Europe/Kiev' })}</i>`, {
parse_mode: 'HTML' });
                                    user(fromID).timesWasBanned =
user(fromID).timesWasBanned + 1;
                                    updateUsers();
                                    const interval = setInterval(() => {
                                        if (new Date().getTime() >= period) {
                                            bot.promoteChatMember(groupID, fromID);
                                            clearInterval(interval);
                                        }
                                    }, 1000);
                                }
                            })
                        }
                    })
                })
        });
        return;
    }
    if (!user(fromID).isAdmin && isBadWord(disallowedWordlist.join(','), msg.text)) {
        bot.deleteMessage(chatID, msg.message_id);
```

```
        if (!user(fromID).warned) {
            bot.sendMessage(chatID, `⚠<b>Попередження!</b>⚠\n<i>В цьому чаті не можна вживати лайливі висловлювання! Наступного разу вас буде заблоковано на 5хв, кожного наступного разу термін блокування буде зростати.</i>`, { parse_mode: 'HTML' });
            user(fromID).warned = true;
            updateUsers();
            return;
        }
        const period = new Date().getTime() + (10000 * user(fromID).timesWasBanned);
        bot.restrictChatMember(chatID, fromID, { until_date: period }).then(response => {
            if (response) {
                bot.sendMessage(chatID, `🚫<b>Юзера ${from.first_name || from.last_name} було заблоковано</b>🚫\n<b>Причина</b>: <i>вживання заборонених слів</i>\n<b>Дата відновлення користуванням чату</b>: <i>${new Date(period).toLocaleString('uk-UA', { timeZone: 'Europe/Kiev' })}</i>`, { parse_mode: 'HTML' });
                user(fromID).timesWasBanned = user(fromID).timesWasBanned + 1;
                updateUsers();
                    if (new Date().getTime() >= period) {
                        bot.promoteChatMember(groupID, fromID);
                        clearInterval(interval);
                    }
                }, 1000);
            }
        })
    }

    if (/[-a-zA-Z0-9@:%_\+.~#?&//=]{2,256}\.[a-z]{2,4}\b(\/[-a-zA-Z0-9@:%_\+.~#?&//=]*)?/gi.test(msg.text)) {
        bot.deleteMessage(chatID, msg.message_id);
        bot.sendMessage(chatID, `⚠<b>В чьому чаті заборонено використовувати посилання</b>⚠`, { parse_mode: 'HTML' });
    }

});
```

ФАЙЛ КОНФІГУРАЦІЇ ТА КЕРУВАННЯ ЗАЛЕЖНОСТЯМИ

Файл package.json

```
{
  "name": "bad-word-check-bot",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "@deepgram/sdk": "^2.1.1",
    "@ffmpeg-installer/ffmpeg": "^1.1.0",
    "fluent-ffmpeg": "^2.1.2",
    "node-telegram-bot-api": "^0.61.0",
    "openai": "^3.2.1"
  }
```

ЛІСТИНГ МОДУЛЯ РОЗІЗНАВАННЯ ГОЛОСОВИХ ПОВІДОМЛЕНЬ

Файл deepgramTranscription.js

```
}
const { Deepgram } = require('@deepgram/sdk');
const fs = require('fs');

function transcribe(path) {
    const deepgramApiKey = '14d439186af7ec1e68a5947357236f9d16bee169';
    const pathToFile = path;
    const mimetype = 'audio/mp3';
    const deepgram = new Deepgram(deepgramApiKey);
    return new Promise((resolve, reject) => {
        const source = {
            buffer: fs.readFileSync(pathToFile),
            mimetype
        };
        const options = {
            punctuate: true,
            model: 'whisper',
            detect_language: true
        };
        deepgram.transcription.preRecorded(source, options)
            .then((transcription) => {
                resolve(transcription.results.channels[0].alternatives[0])
                fs.unlinkSync(path);
                fs.unlinkSync(path.replace('.mp3', '.oga'));
            })
            .catch((err) => {
                console.log(err);
            });
    });
}

module.exports = transcribe;
```

МОДУЛЬ ВИЯВЛЕННЯ ЗАБОРОНЕНИХ СЛІВ У ПОВІДОМЛЕННЯХ

Файл detectDisallowedWords.js

```
class detectDisallowedWords {
    constructor(props) {
        if (props && typeof props.dictionary === 'string') {
            this.dictionary = props.dictionary.replace(/\s+/g,
'').toLowerCase().split(',');
        } else {
            this.dictionary = null;
        }
    }
    check = (string) => {
        if (typeof string !== 'string') {
            return console.error('Відсутній текст для перевірки!');
        }
        if (!this.dictionary) {
            return console.error('Відсутній словник заборонених слів!');
        }
        const allWords = [...this.dictionary];
        return string.toLowerCase().match(new RegExp(allWords.join('|'))) !== null;
    }
}
module.exports = detectDisallowedWords;
const detectDisallowedWords = require('../controllers/detectDisallowedWords');
function isBadWord(array, string) {
    const filter = new detectDisallowedWords({
        dictionary: array
    });
    return filter.check(string);
}

module.exports = isBadWord;
const ffmpeg = require('fluent-ffmpeg');
 ('@ffmpeg-installer/ffmpeg').path;
const fs = require('fs');
function converter(inputAudioPath, outputAudioPath) {
    ffmpeg.setFfmpegPath(ffmpegPath);
    const outStream = fs.createWriteStream(outputAudioPath);
    return new Promise((resolve, reject) => {
        ffmpeg()
        .input(inputAudioPath)
        .audioQuality(96)
        .toFormat("mp3")
        .on('error', error => reject(`Encoding Error: ${error.message}`))
        .on('exit', () => console.log('Audio recorder exited'))
        .on('close', () => console.log('Audio recorder closed'))
```

```
        .on('end', () => resolve(outputAudioPath))
        .pipe(outStream, { end: true });
}

module.exports = converter;

const { Configuration, OpenAIApi } = require("openai");
const fs = require('fs');
const config = new Configuration({
    apiKey: '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',
});

async function transcribe(path) {
    const openai = new OpenAIApi(config);
    const response = await openai.createTranscription(
        fs.readFileSync(path).toString('base64'),
        "whisper-1",
    ).then(res => {
        console.log(res);
    })
    .catch(error => {
    })

    return response;
}
```

Державний університет інформаційно-комунікаційних телекомунікацій

Навчально-науковий інститут інформаційних технологій
кафедра інформаційних систем та технологій

Магістерська дисертація
на здобуття ступеня магістра за освітньо-професійною програмою
"Інтелектуальні системи управління" зі спеціальності *124 "Системний аналіз"*

на тему: **«Система моніторингу інформаційної безпеки при використанні корпоративних чатів у Telegram»**

Виконав: студент 2 курсу, групи САДМ-61 **ЧЕРЕВАТИЙ Богдан Сергійович**

Керівник: *професор, д.т.н., професор* ***Шушура Олексій Миколайович***

АКТУАЛЬНІСТЬ

- корпоративна комунікація масово переходить у месенджери, зокрема Telegram;
- Telegram активно використовується для службових та конфіденційних даних;
- виникають ризики витоку інформації, зловживань та несанкціонованого доступу;
- в Telegram немає вбудованого централізованого контролю політик безпеки;
- потрібна спеціалізована система моніторингу безпеки корпоративних чатів Telegram.

МЕТА ТА ЗАВДАННЯ РОБОТИ

Мета роботи: розробити систему моніторингу, призначену для виявлення та запобігання порушенням інформаційної безпеки в корпоративних чатах месенджеру Telegram.

Завдання роботи:

– проаналізувати особливості корпоративної комунікації та механізми Telegram у контексті інформаційної безпеки;

– виявити загрози існуючих підходів до захисту корпоративних чатів;

– сформулювати вимоги до системи моніторингу інформаційної архітектури чатів Telegram;

– розробити концептуальну модель та архітектуру системи;

– реалізувати програмний прототип системи моніторингу.

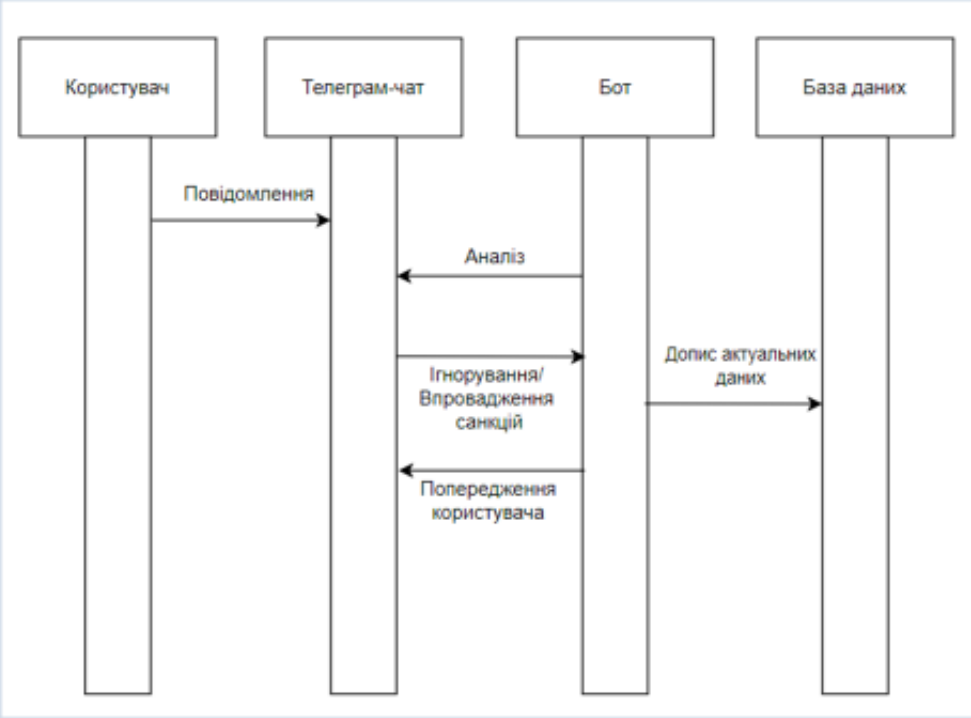
ОБ’ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

Об’єкт дослідження: процес забезпечення інформаційної безпеки в корпоративних комунікаціях.

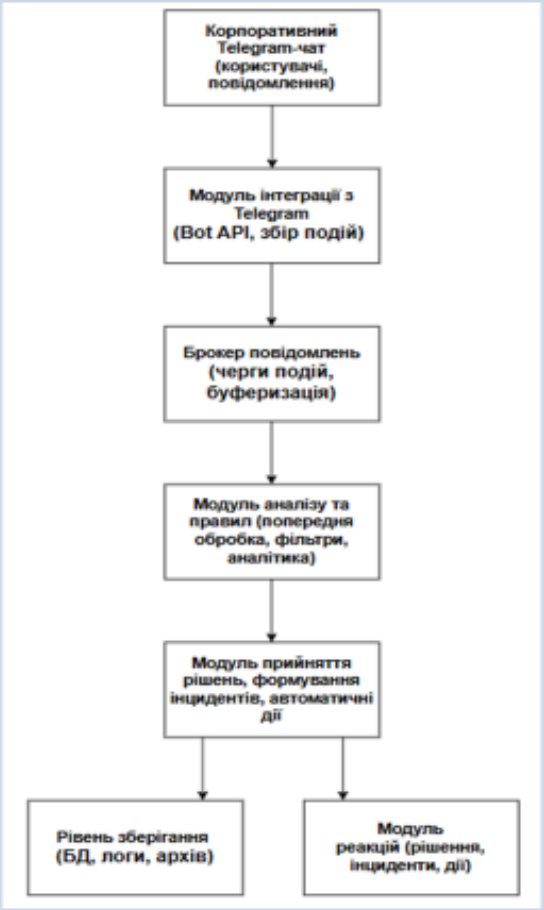
Предмет дослідження: методи, моделі та програмні засоби моніторингу інформаційної безпеки в корпоративних чатах Telegram.

5

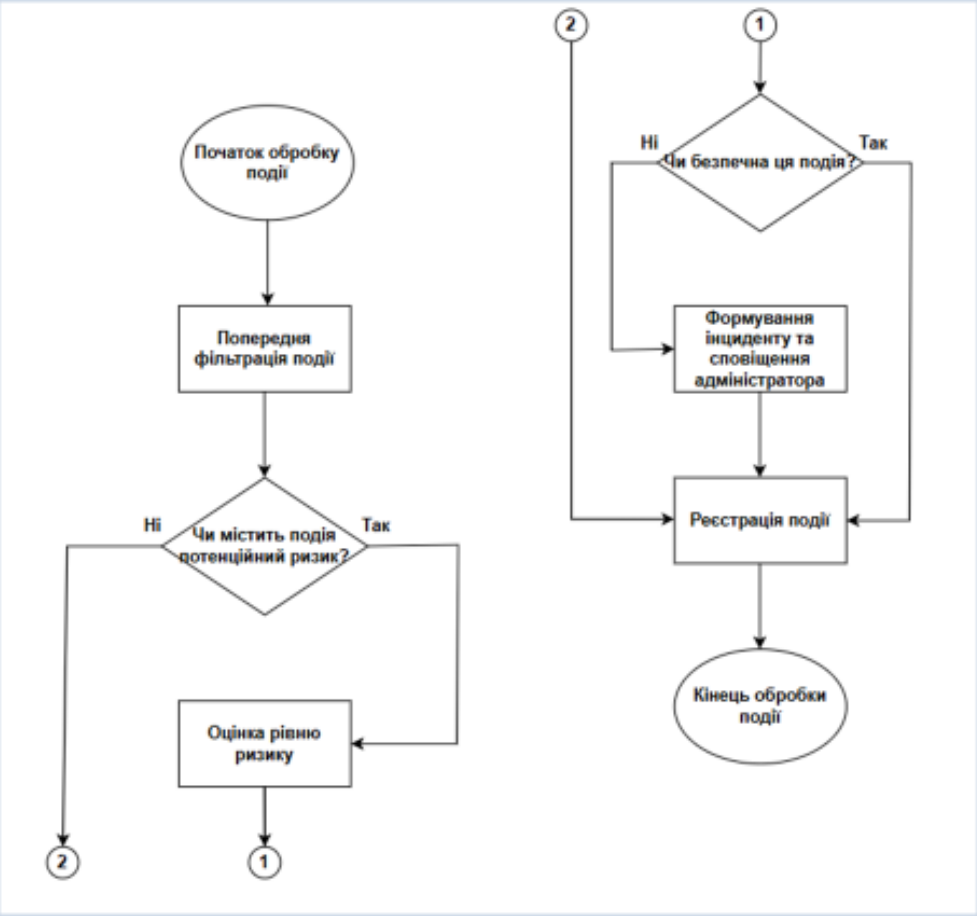
ДІАГРАМА ПОСЛІДОВНІСТЬ ДІЙ МОДЕЛІ БОТУ TELEGRAM



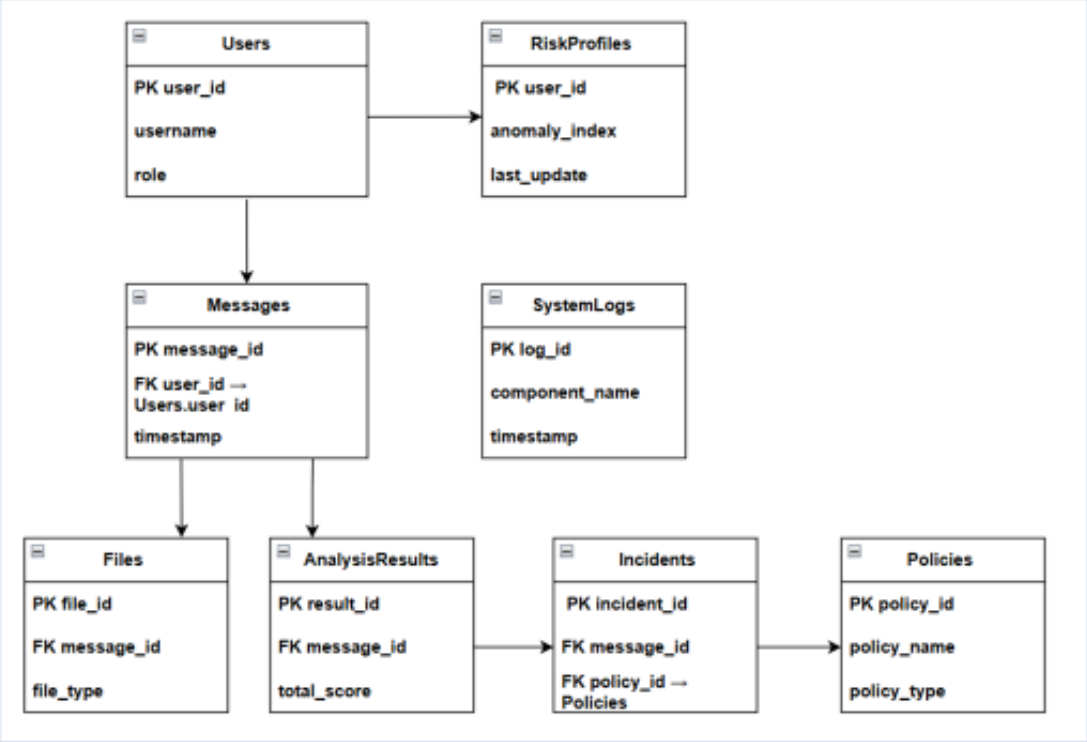
ДІАГРАМА АРХІТЕКТУРИ ПРОГРАМНОЇ СИСТЕМИ МОНІТОРИНГУ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ



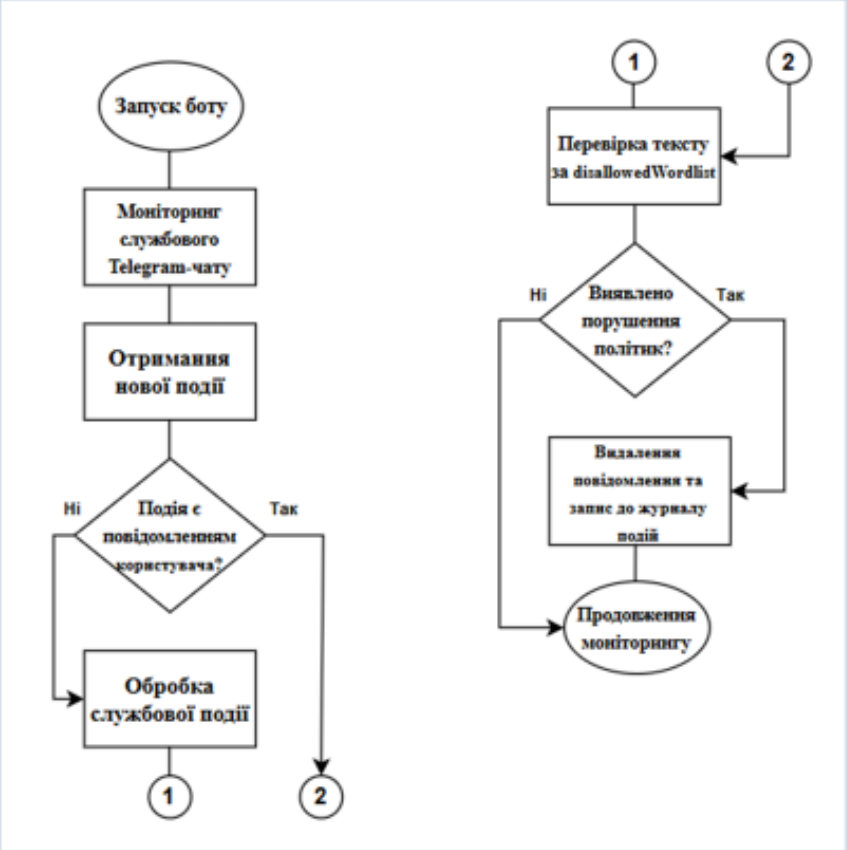
АЛГОРИТМ ОБРОБКИ ПОДІЙ ТА ВИЯВЛЕННЯ ПОРУШЕНЬ ПОЛІТИК БЕЗПЕКИ



КОНЦЕПТУАЛЬНА МОДЕЛЬ БАЗИ ДАНИХ СИСТЕМИ МОНІТОРИНГУ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ



АЛГОРИТМ РОБОТИ БОТУ TELEGRAM ДЛЯ МОНІТОРИНГУ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ

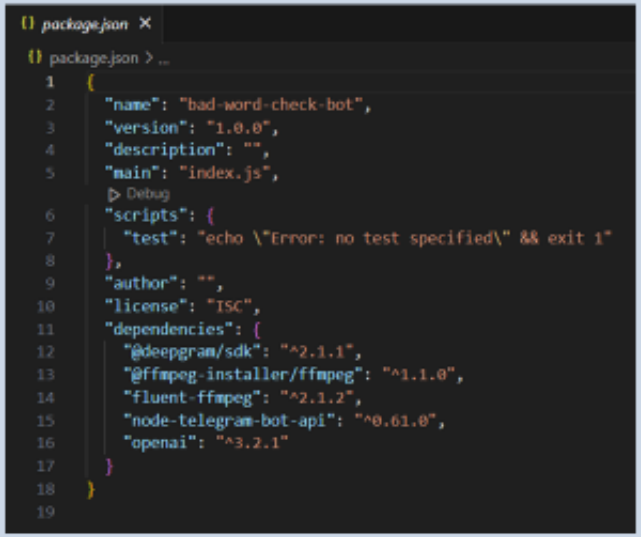


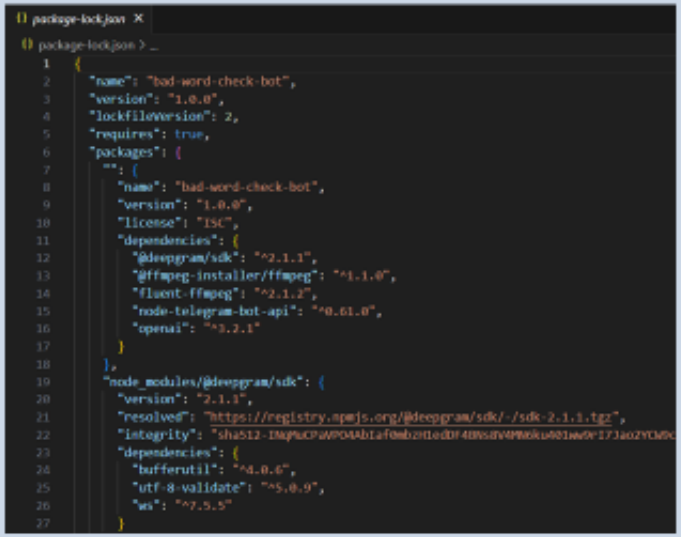
10

ІНСТРУМЕНТИ ДЛЯ РЕАЛІЗАЦІЇ

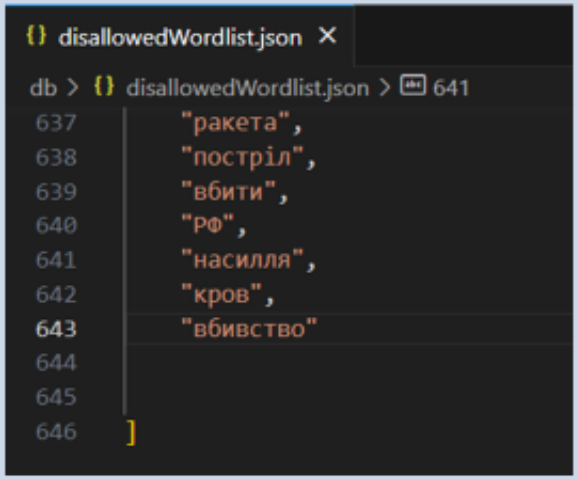
TELEGRAM BOT API

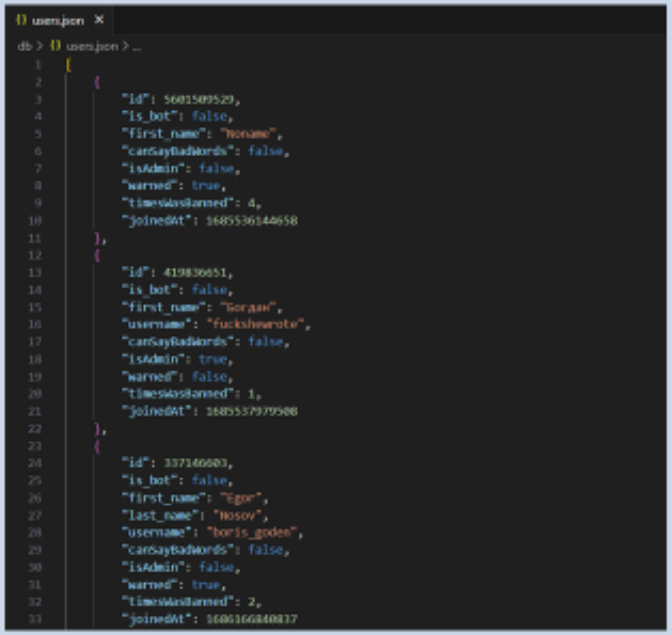
ФАЙЛИ З ОПИСОМ КОНФІГУРАЦІЇ ТА ЗАЛЕЖНОСТЕЙ БОТУ TELEGRAM



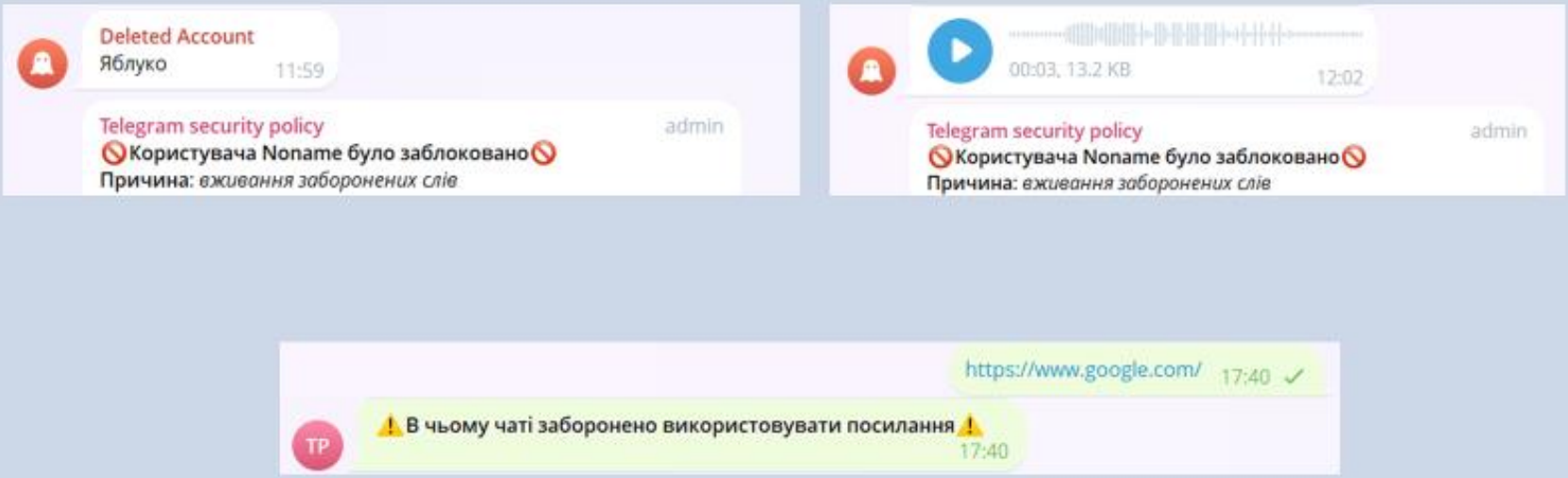


ФРАГМЕНТ ФАЙЛІВ З СЛОВНИКОМ ЗАБОРОНЕНИХ СЛІВ ТА ЗБЕРІГАННЯ ІНФОРМАЦІЇ ПРО КОРИСТУВАЧІВ ЧАТУ TELEGRAM





ІНТЕРФЕЙС РОБОТИ БОТУ TELEGRAM



ВИСНОВКИ

У магістерській роботі розроблено систему моніторингу інформаційної безпеки корпоративних чатів у Telegram, що забезпечує контроль дотримання політик, виявлення порушень та формування інцидентів із сповіщенням адміністратора. На основі аналізу корпоративної комунікації, загроз та існуючих підходів побудовано концепцію системи, розроблено її модель, архітектуру, структуру бази даних і алгоритми аналізу тексту, файлів і поведінки користувачів. Реалізовано прототип Telegram-бота, який пройшов тестування на типових сценаріях і продемонстрував здатність зменшувати потребу в ручній модерації та може бути використаний як основа для подальшого впровадження й розвитку системи моніторингу в реальних корпоративних умовах.

АПРОБАЦІЯ

1. Череватий Б.С., Шушура О.М. Система моніторингу інформаційної безпеки при використанні корпоративних чатів у Telegram. Матеріали III Всеукраїнської науково-технічної конференції «Технологічні горизонти: дослідження та застосування інформаційних технологій для технологічного прогресу України і світу», 18 листопада 2025 року.
2. Череватий Б.С., Шушура О.М., Соломаха С.А., Система забезпечення моніторингу інформаційної безпеки корпоративних Telegram-чатів. Матеріали VIII Всеукраїнської науково-технічної конференції «Комп'ютерні технології: інновації, проблеми, рішення», м. Житомир, 02-03 грудня 2025 року.

ДЯКУЮ ЗА УВАГУ