МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВОЛИНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ЛЕСІ УКРАЇНКИ
Кафедра комп'ютерних наук та кібербезпеки

ХМІЛЕВСЬКА АЛІНА ВАСИЛІВНА

РОЗРОБКА КОМПЛЕКСНОЇ СИСТЕМИ ЗАХИСТУ ІНФОРМАЦІЇ НА МОБІЛЬНИХ ПРИСТРОЯХ З ФУНКЦІЄЮ ВІДДАЛЕНОГО ВИДАЛЕННЯ ДАНИХ

Спеціальність: 122 Комп'ютерні науки
Освітньо-професійна програма: Комп'ютерні науки та інформаційні технології
Кваліфікаційна робота на здобуття освітнього ступеня «бакалавр»

Науковий керівник:
БУЛАТЕЦЬКА ЛЕСЯ ВІТАЛІЇВНА,
кандидат фізико-математичних наук, доцент кафедри
комп'ютерних наук та кібербезпеки

РЕКОМЕНДОВАНО ДО ЗАХИСТУ
Протокол №_____
засідання кафедри комп'ютерних наук
та кібербезпеки
від _____ 2025 р.
Завідувач кафедри
(_____) Гришанович Т. О.

ЛУЦЬК - 2025

ЗМІСТ

ВСТУП

Мобільні пристрої стали ключовим елементом цифрової інфраструктури, через яку проходить величезна кількість особистих, фінансових, робочих та конфіденційних даних. Смартфони вже давно не обмежуються лише функціями комунікації – вони виступають одночасно як фотоархіви, банківські термінали, засоби ідентифікації, доступу до корпоративних мереж та персональних хмарних сховищ. Через це зростає ризик несанкціонованого доступу до інформації в разі втрати пристрою, крадіжки, злому або компрометації.

Вразливість мобільних пристроїв обумовлена як технічними, так і людськими факторами: використанням слабких паролів, відсутністю оновлень, неправильними налаштуваннями дозволів, а також зростанням кількості шкідливого ПЗ і фішингових атак. Особливо гостро проблема постає в умовах, коли користувач фізично втрачає контроль над пристроєм. У таких випадках ефективним способом захисту може стати механізм віддаленого видалення даних.

Актуальність теми полягає у потребі забезпечення цілісного захисту інформації на мобільних пристроях із можливістю швидкого реагування на інциденти. Наявні на ринку рішення не завжди відповідають вимогам безперебійної роботи в автономному режимі або підтримки вибіркового видалення. Тому розробка комплексної системи, яка поєднує багаторівневий захист, незалежність від зовнішніх факторів та можливість безповоротного видалення критичних даних, є важливим напрямом дослідження.

Мета роботи – розробити систему захисту інформації для Android-пристроїв, що включає функцію віддаленого видалення даних та здатна працювати в умовах нестабільного або відсутнього підключення до мережі.

Для досягнення цієї мети потрібно:

- проаналізувати існуючі загрози безпеці даних на мобільних пристроях;
- дослідити сучасні методи захисту інформації;

- вивчити принципи роботи механізмів віддаленого управління пристроями на Android;
- розробити архітектуру системи, що забезпечує гарантоване видалення даних;
- реалізувати прототип з використанням сучасного технологічного стеку (Flutter, Firebase, Kotlin);
- забезпечити безпечну обробку команд та захист від несанкціонованого доступу;
- провести тестування на різних пристроях і в умовах обмеженого зв’язку;
- сформулювати рекомендації з використання системи та можливості її розвитку.

Об’єкт дослідження – засоби забезпечення інформаційної безпеки на мобільних пристроях.

Предмет дослідження – методи, алгоритми та архітектурні рішення для реалізації функції віддаленого видалення даних на платформі Android.

Апробація результатів та публікації.

Хмілевська А., Булатецька Л. Розробка системи віддаленого видалення даних на android-пристроях як елемента мобільної безпеки. Проблеми комп’ютерних наук, програмного моделювання та безпеки цифрових систем.: матеріали II міжнар. науково-практ. конф. Луцьк, 9–11 червн. 2025 р. Луцьк, 2025.

РОЗДІЛ 1
АНАЛІЗ МЕТОДІВ ЗАХИСТУ ІНФОРМАЦІЇ ТА ВІДДАЛЕНОГО ВИДАЛЕННЯ ДАНИХ НА МОБІЛЬНИХ ПРИСТРОЯХ

1.1 Поняття конфіденційної інформації та методи її захисту на мобільних пристроях

У сучасному інформаційному суспільстві мобільні пристрої стали невід'ємною частиною повсякденного життя та професійної діяльності людей. Вони використовуються не лише для комунікації, але й для зберігання та обробки значних обсягів конфіденційної інформації. Згідно з визначенням Закону України "Про інформацію", конфіденційна інформація – це відомості, які знаходяться у володінні, користуванні або розпорядженні окремих фізичних чи юридичних осіб і поширюються за їх бажанням відповідно до передбачених ними умов [1].

На мобільних пристроях конфіденційна інформація може включати особисті дані користувача, фінансову інформацію, корпоративні документи, медичні записи, паролі та ключі доступу, а також дані про місцезнаходження та активність користувача. Особливої актуальності питання захисту такої інформації набуває в умовах зростання кількості кібератак та інцидентів, пов'язаних з витоком даних.

За даними Національного інституту стандартів і технологій США (NIST), кількість інцидентів безпеки, пов'язаних з мобільними пристроями, щороку зростає на 15-20% [2]. При цьому, згідно з дослідженням компанії Verizon, представленим у звіті "Data Breach Investigations Report", близько 43% випадків витоку даних пов'язані саме з мобільними пристроями [3]. Ці статистичні дані підкреслюють важливість розробки та впровадження ефективних методів захисту конфіденційної інформації на мобільних пристроях.

Для захисту конфіденційної інформації на мобільних пристроях застосовуються різноманітні методи та технології, які можна класифікувати за кількома основними напрямами (табл. 1.1).

Таблиця 1.1

Основні методи захисту конфіденційної інформації на мобільних пристроях

Категорія методів	**Опис**	**Приклади реалізації**
Технології шифрування	Перетворення даних у нечитабельний формат, який можна відновити лише за допомогою відповідного ключа	Повне шифрування пристрою, шифрування окремих файлів та додатків
Механізми аутентифікації	Перевірка особи користувача для надання доступу до пристрою та даних	Паролі, PIN-коди, графічні ключі, біометрична аутентифікація
Технології управління доступом	Контроль рівнів доступу до різних типів даних та функцій пристрою	Дозволи додатків, системи контролю доступу на основі ролей
Системи віддаленого управління	Дистанційний контроль над пристроєм та його даними	Відстеження місцезнаходження, віддалене блокування, віддалене видалення даних
Безпечне зберігання даних	Забезпечення безпеки даних, які зберігаються на пристрої	Захищені сховища, ізольовані контейнери, безпечні елементи

Шифрування даних є фундаментальним методом захисту конфіденційної інформації на мобільних пристроях. Згідно з рекомендаціями Міжнародної організації зі стандартизації (ISO/IEC 27001), ефективне шифрування повинно забезпечувати конфіденційність, цілісність та автентичність даних [4]. На сучасних мобільних пристроях найчастіше використовуються симетричні алгоритми шифрування, такі як Advanced Encryption Standard (AES) з довжиною ключа 256 біт, який на сьогоднішній день вважається стійким до сучасних методів криптоаналізу.

Дослідження, проведене Інститутом комп'ютерної безпеки (Computer Security Institute), показує, що впровадження повного шифрування пристрою зменшує ризик компрометації даних при втраті або крадіжці пристрою на 85-90% [5]. Однак, важливо зазначити, що ефективність шифрування значною мірою залежить від надійності ключів шифрування, які, у свою чергу, часто базуються на паролях користувачів.

Механізми аутентифікації відіграють важливу роль у захисті конфіденційної інформації, оскільки вони забезпечують перший рівень захисту від несанкціонованого доступу до пристрою. Традиційні методи аутентифікації, такі як паролі та PIN-коди, сьогодні доповнюються більш сучасними та зручними методами, зокрема біометричною аутентифікацією.

Технології управління доступом забезпечують контроль над тим, які програми та процеси мають доступ до конфіденційної інформації на пристрої. У системі Android реалізовано модель дозволів, яка вимагає від додатків явного запиту на доступ до різних типів даних та функцій пристрою. Починаючи з Android 6.0 (Marshmallow), користувачі можуть надавати або відкликати ці дозволи в будь-який момент, що підвищує гнучкість та безпеку системи.

Для корпоративних середовищ важливим інструментом захисту конфіденційної інформації є системи контролю доступу на основі ролей (Role-Based Access Control, RBAC), які обмежують доступ до даних залежно від ролі користувача в організації. Згідно з дослідженням, проведеним Ponemon Institute, впровадження RBAC зменшує ризик внутрішніх витоків даних на 60-70% [6].

Системи віддаленого управління дозволяють контролювати пристрій та його дані навіть у випадку фізичної втрати контролю над пристроєм. Ключовою функцією таких систем є можливість віддаленого видалення даних (Remote Wipe), яка забезпечує безповоротне знищення конфіденційної інформації на пристрої у разі його втрати або крадіжки.

Для реалізації функції віддаленого видалення даних на платформі Android використовуються різні технології, зокрема Firebase Cloud Messaging для передачі команд на пристрій та Android Device Administration API для виконання

операцій видалення [7]. Важливою особливістю сучасних рішень є можливість виконання цих операцій навіть у випадку, коли пристрій не має активного підключення до Інтернету в момент ініціації команди – вони виконуються при першому підключенні пристрою до мережі.

Дослідження, проведене Security Research Labs, показало, що час від ініціації команди віддаленого видалення до її виконання на пристрої може варіюватися від кількох секунд до кількох годин, залежно від стану підключення пристрою до мережі та налаштувань енергозбереження [8]. Для критично важливих даних рекомендується використовувати додаткові методи захисту, такі як шифрування та контейнеризація, які забезпечують безпеку даних навіть у випадку затримки виконання команди видалення.

Безпечне зберігання даних на мобільних пристроях забезпечується за допомогою спеціалізованих сховищ та контейнерів, які ізолюють конфіденційну інформацію від інших даних на пристрої. У системі Android для цього використовуються технології, такі як Android Keystore System, який забезпечує безпечне зберігання криптографічних ключів, та Secure Element, який представляє собою апаратно захищений модуль для зберігання особливо чутливих даних.

Ефективний захист конфіденційної інформації на мобільних пристроях вимагає комплексного підходу, який поєднує різні методи та технології, створюючи багаторівневу систему захисту (рис. 1.1). Згідно з рекомендаціями Національного інституту стандартів і технологій США, така система повинна включати не лише технічні засоби захисту, але й організаційні заходи, такі як розробка та впровадження політик безпеки, навчання користувачів та регулярний аудит безпеки [9].

Варто зазначити, що ефективність методів захисту конфіденційної інформації на мобільних пристроях значною мірою залежить від їх правильної реалізації та використання. Навіть найбільш надійні технології шифрування можуть бути скомпрометовані при використанні слабких паролів або наявності вразливостей у програмному забезпеченні.

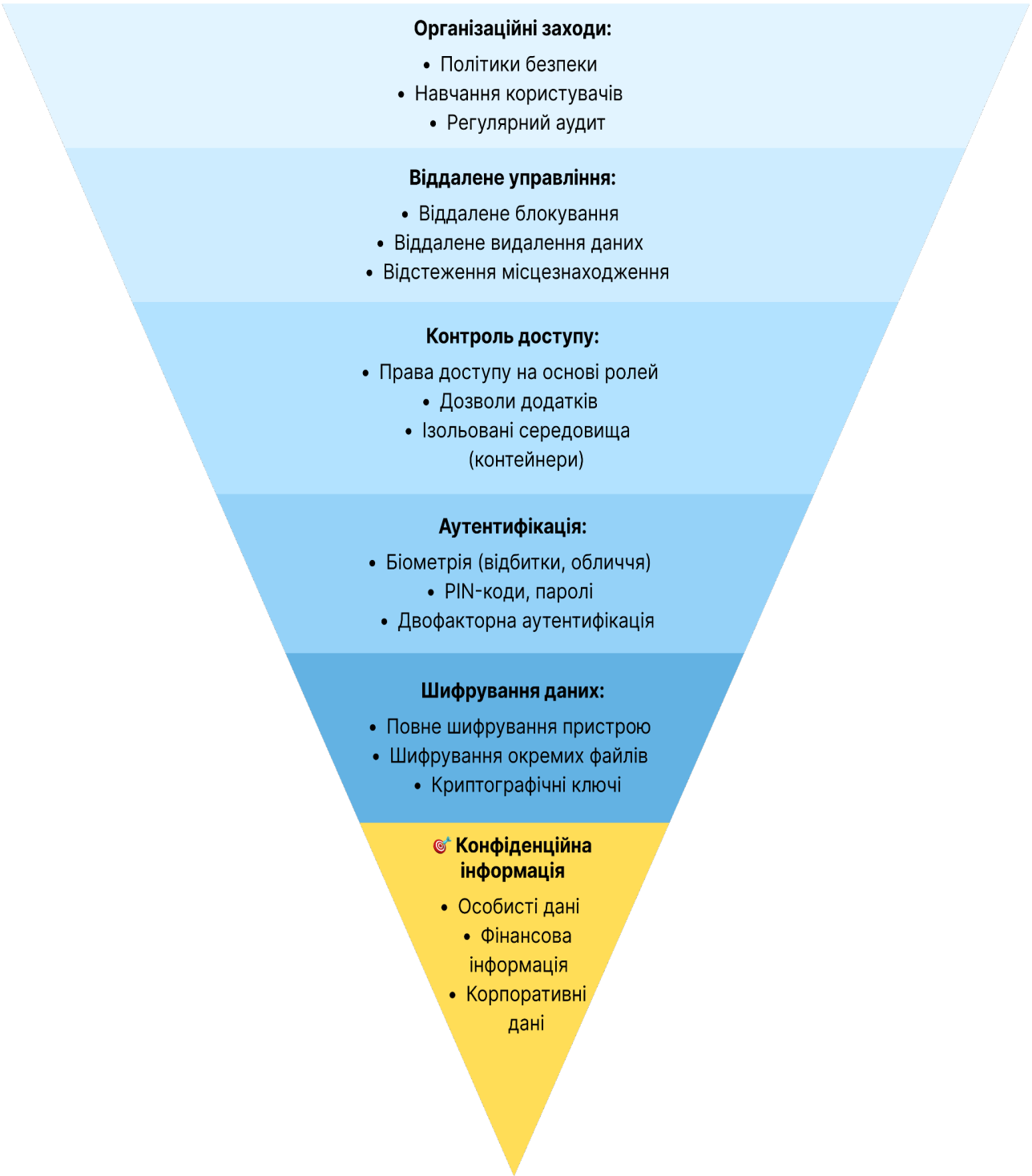

Рисунок 1.1 - Схема багаторівневого захисту конфіденційної інформації на мобільних пристроях

Таким чином, захист конфіденційної інформації на мобільних пристроях є комплексним завданням, яке вимагає застосування різних методів та технологій, адаптованих до конкретних умов використання пристрою та характеру інформації, що захищається. Проте, незважаючи на існування численних засобів

захисту, сучасні кіберзагрози постійно еволюціонують, створюючи нові виклики для систем безпеки.

1.2 Аналіз сучасних загроз безпеці даних на мобільних пристроях

У сучасному цифровому середовищі мобільні пристрої перетворилися на універсальні інструменти для роботи, комунікації, банківських операцій, зберігання документів та особистих спогадів. Водночас їх багатофункціональність, постійне підключення до Інтернету та висока мобільність роблять їх вразливими до багатьох типів загроз. Ступінь ризику значно зростає у випадках використання пристроїв у корпоративному або військовому середовищі, де мова йде не лише про втрату особистих даних, а й про ймовірний витік конфіденційної чи стратегічної інформації.

Загрози безпеці даних на мобільних пристроях можна класифікувати за різними критеріями. На основі джерела походження та механізму реалізації загрози найбільш доцільним є поділ на чотири основні категорії (табл. 1.2.).

Таблиця 1.2

Основні категорії загроз безпеці даних на мобільних пристроях

Категорія загроз	**Опис**	**Приклади**
Програмні загрози	Загрози, пов’язані з шкідливим програмним забезпеченням та вразливостями в ПЗ	Віруси, троянські програми, шпигунське ПЗ, програми-вимагачі
Апаратні загрози	Загрози, пов’язані з фізичним доступом до пристрою	Крадіжка або втрата пристрою, клонування SIM-карт, апаратні модифікації
Мережеві загрози	Загрози, пов’язані з передачею даних через мережі	Атаки "людина посередині", перехоплення даних у незахищених Wi-Fi мережах
Соціоінженерні загрози	Загрози, що використовують людський фактор	Фішинг, вішинг (голосовий фішинг), смішинг (SMS-фішинг)

Програмні загрози охоплюють шкідливе програмне забезпечення, здатне інфікувати пристрій та виконувати несанкціоновані дії. Згідно зі статистикою лабораторії Kaspersky, у 2024 році зафіксовано понад 33,3 мільйона атак на користувачів мобільних пристроїв з використанням різних типів шкідливого ПЗ, а кількість атак банківських троянів на смартфони зросла на 196% порівняно з попереднім роком [10]. Найпоширенішим типом загроз є рекламне ПЗ (adware), що складає близько 46% всіх виявлених загроз на мобільних пристроях. Це програми, які відображають нав’язливу рекламу та збирають дані про користувача. Банківські трояни, спрямовані на крадіжку облікових даних для онлайн-банкінгу, також становлять значну загрозу.

Шпигунське ПЗ (spyware) таємно збирає інформацію про користувача, включаючи повідомлення, контакти, дані про місцезнаходження та іншу конфіденційну інформацію, тоді як програми-вимагачі (ransomware) шифрують дані на пристрої та вимагають викуп за їх розшифрування. Кіберзлочинці часто поширюють мобільні загрози через офіційні та неофіційні магазини додатків. У 2023-2024 роках спостерігалися численні випадки проникнення шкідливих додатків у Google Play (рис. 1.2). Фальшиві інвестиційні додатки часто використовують соціальну інженерію для отримання особистих даних користувачів, які пізніше додаються до баз даних для телефонного шахрайства.

Фізичні загрози, такі як втрата або крадіжка пристрою, залишаються одним із найчастіших сценаріїв компрометації даних. Без належного захисту (шифрування, аутентифікації, автоматичного блокування екрана) зловмисник може швидко отримати доступ до контактів, фото, листування або службових файлів. У критичних умовах — наприклад, у зоні бойових дій або під час роботи журналістів — такий витік може мати серйозні наслідки для безпеки особи чи держави. За даними дослідження, проведеного Ponemon Institute, значна частка організацій повідомляють про втрату або крадіжку мобільних пристроїв, що містять конфіденційну інформацію, протягом останнього року. При цьому, менше половини організацій використовують рішення для віддаленого

видалення даних, які могли б мінімізувати ризики, пов’язані з фізичною втратою контролю над пристроєм.

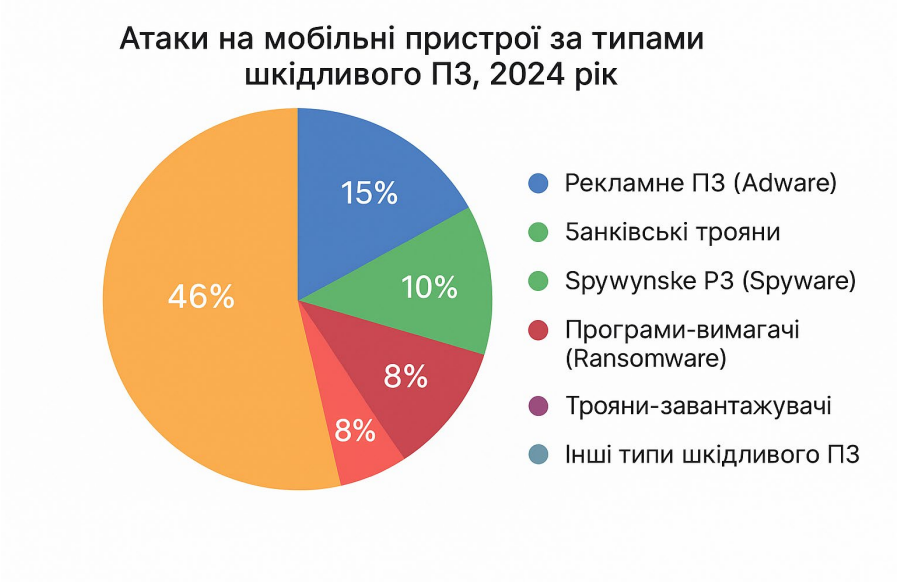

Рисунок 1.2 - Розподіл атак на мобільні пристрої за типами шкідливого ПЗ у 2024 році

Особливу увагу варто звернути на мережеві загрози, які реалізуються через публічні Wi-Fi мережі або підміни точок доступу. Атаки типу "людина посередині" (Man-in-the-Middle, MitM) дозволяють перехоплювати незашифровану інформацію або підміняти трафік. Дослідники з Check Point повідомляють про зростання кількості атак через фейкові мобільні точки доступу, які імітують легітимні мережі, особливо у громадських місцях [11]. Wi-Fi-спуфінг, або створення підроблених точок доступу, які імітують легітимні мережі, є ефективним способом перехоплення даних користувачів.

Соціальна інженерія є одним із найефективніших векторів атаки на користувачів мобільних пристроїв. Фішинг, вішинг (голосовий фішинг) та смішинг (SMS-фішинг) базуються на психологічному впливі на користувача з метою змусити його добровільно розкрити паролі, підтвердити доступ до банківських рахунків або встановити шкідливий застосунок. За даними Verizon’s 2023 Data Breach Investigations Report, переважна більшість кібератак мотивовані фінансовою вигодою, а соціальна інженерія використовується у значній частині

успішних атак. Особливу загрозу становить використання штучного інтелекту для створення більш переконливих фішингових повідомлень. Згідно з прогнозом Gartner, у найближчі роки значна частка підприємств вважатимуть рішення для перевірки особи ненадійними через створені за допомогою ШІ діпфейки [12].

Аналіз статистичних даних за останні роки показує стійке зростання як кількості, так і складності атак на мобільні пристрої (рис. 1.3).

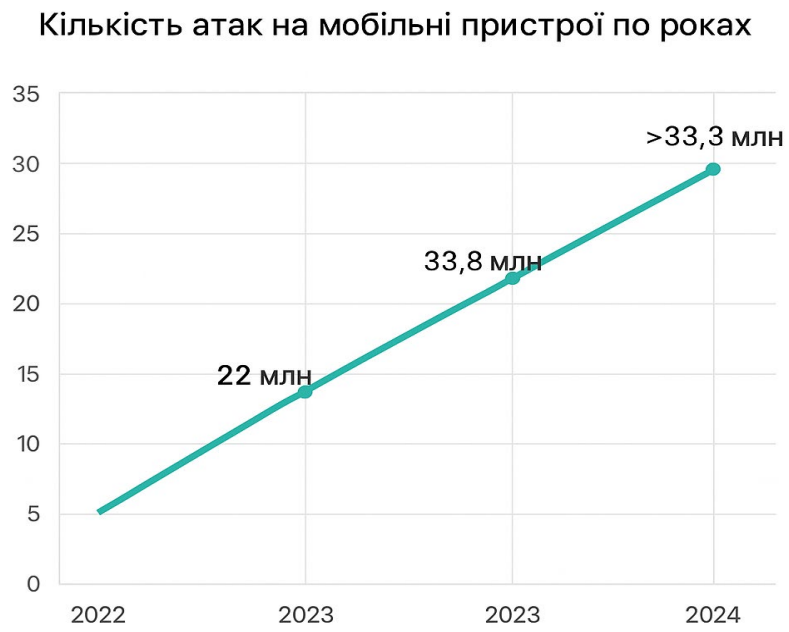

Рисунок 1.3 - Динаміка атак на мобільні пристрої за 2022-2024 роки

Вплив загроз на безпеку даних на мобільних пристроях суттєво відрізняється залежно від категорії користувачів та характеру інформації, що зберігається на пристрої. Індивідуальні користувачі ризикують втратою особистих даних, фінансової інформації, доступу до соціальних мереж та електронної пошти. Корпоративні користувачі стикаються з загрозами не лише щодо особистих даних, але й щодо корпоративної інформації, включаючи комерційну таємницю, конфіденційні документи, дані клієнтів та фінансову інформацію. Втрата або компрометація таких даних може призвести до значних фінансових втрат та репутаційних ризиків для компанії. Для державних та

військових структур компрометація даних може мати наслідки для національної безпеки, тому мобільні пристрої працівників таких структур часто є ціллю цілеспрямованих атак, спонсорованих державами або організованими злочинними групами.

У контексті описаних загроз особливого значення набувають системи віддаленого управління та видалення даних, які дозволяють контролювати пристрої та знищувати конфіденційну інформацію у випадку компрометації. Такі системи є критично важливими для корпоративних та державних організацій, які працюють з чутливою інформацією. Основні сценарії використання систем віддаленого управління включають втрату або крадіжку пристрою, компрометацію облікових даних, звільнення працівника та надзвичайні ситуації..

Подальший розвиток та ефективне впровадження механізмів віддаленого управління мобільними пристроями на базі Android є логічним продовженням розвитку систем захисту конфіденційної інформації. З огляду на різноманітність та складність сучасних загроз, віддалене управління пристроями стає не просто додатковим засобом захисту, а необхідним компонентом комплексної стратегії безпеки. Це зумовлює необхідність детального вивчення технологій віддаленого управління Android-пристроями та принципів їх роботи.

1.3 Технології віддаленого управління Android-пристроями: огляд та принципи роботи

З розвитком екосистеми Android та поширенням мобільних пристроїв у корпоративному та державному секторах, компанія Google розробила ряд технологій для віддаленого управління мобільними пристроями, які еволюціонували з часом. Першим значним кроком став Device Administration API, представлений у Android 2.2 як базовий механізм віддаленого управління, що дозволяв адміністраторам встановлювати політики паролів, блокувати пристрої та виконувати віддалене видалення даних. Пізніше з’явився Android

Enterprise (раніше Android for Work) - більш розширений набір інструментів, представлений у Android 5.0 (Lollipop), який забезпечив створення робочих профілів та більш гнучке управління застосунками. Сучасним інструментом став Android Management API - програмний інтерфейс, що спрощує процес розробки рішень Mobile Device Management (MDM).

Сучасна система віддаленого управління Android-пристроями має складну архітектуру, що включає кілька ключових компонентів (рис. 1.4). Центральним елементом є EMM-сервер (Enterprise Mobility Management), який забезпечує інтерфейс адміністрування, зберігає політики безпеки та керує процесом комунікації з пристроями. Взаємодія відбувається через API-сервіси Google - програмні інтерфейси, що забезпечують комунікацію між EMM-сервером та пристроями, насамперед через Android Management API та Firebase Cloud Messaging API. На пристрої працює DPC (Device Policy Controller) - компонент, відповідальний за забезпечення політик безпеки та виконання команд від EMM-сервера. Важливу роль відіграють канали зв’язку, особливо Firebase Cloud Messaging для доставки push-повідомлень.

Рисунок 1.4 - Архітектура системи віддаленого управління Android-пристроями

Процес взаємодії між компонентами системи відбувається послідовно та злагоджено. Адміністратор через EMM-консоль встановлює політики безпеки та надсилає команди, такі як команда видалення даних. EMM-сервер передає ці команди через Android Management API, після чого API-сервіси Google доставляють їх на цільовий пристрій через Firebase Cloud Messaging. DPC на пристрої отримує та виконує команди відповідно до встановлених політик безпеки, а результати виконання передаються назад на EMM-сервер. Така архітектура забезпечує надійний та безпечний механізм віддаленого управління, включаючи функції віддаленого видалення даних.

Device Administration API був першим рішенням для віддаленого управління Android-пристроями. Згідно з документацією Android для розробників, його основними функціями є управління паролями, блокування пристрою, віддалене видалення даних, відстеження невдалих спроб входу та обмеження використання камери. Процес розробки додатку для адміністрування пристроїв з використанням цього API включає створення підкласу DeviceAdminReceiver, декларацію політик безпеки в XML-файлі метаданих, реєстрацію додатку як адміністратора пристрою та реалізацію функцій для керування пристроєм. Віддалене видалення даних здійснюється через виклик методу wipeData() класу DevicePolicyManager, який виконує повне скидання пристрою до заводських налаштувань.

Однак Device Administration API має суттєві обмеження, зокрема відсутність гнучких опцій для вибіркового видалення даних, неможливість розділення особистих та корпоративних даних на пристрої та обмежені можливості управління додатками. Через ці недоліки, починаючи з Android 9, Google поступово обмежує функціональність цього API та рекомендує використовувати Android Enterprise та Android Management API для нових розробок.

Android Enterprise та Android Management API забезпечують більш широкі можливості управління порівняно з Device Administration API [13]. Android Management API надає потужний та гнучкий інтерфейс для керування

пристроями та додатками, включаючи різноманітні опції віддаленого видалення даних. Згідно з документацією Google, цей API дозволяє виконувати повне видалення даних (wipeDevice) з factory reset пристрою, вибіркове видалення даних (wipeWorkProfile) лише робочого профілю з корпоративними даними та налаштування додаткових параметрів видалення, таких як очищення зовнішньої пам’яті чи eSIM.

Порівняння різних підходів до віддаленого видалення даних представлено в таблиці (табл. 1.3). За рівнем безпеки найвищий показник має повне видалення даних, високий – вибіркове видалення, середній – видалення на рівні додатків. Щодо впливу на особисті дані, при повному видаленні втрачаються всі дані, при вибірковому та на рівні додатків особисті дані зберігаються. Повне видалення даних підтримується моделями Fully Managed та COPE, вибіркове – Work Profile та COPE, а на рівні додатків – усіма моделями. Технічна реалізація різниться: повне видалення використовує Device Admin API або Android Management API, вибіркове – Android Management API, а на рівні додатків – Android Enterprise API. Відновлення після видалення при повному підході неможливе без резервного копіювання, а при вибірковому та на рівні додатків можливе для особистих даних.

Таблиця 1.3

Порівняння підходів до віддаленого видалення даних на Android-пристроях

Критерій	**Повне видалення даних**	**Вибіркове видалення даних**	**Видалення даних на рівні додатків**
Рівень безпеки	Найвищий	Високий	Середній
Вплив на особисті дані	Видаляються всі дані	Особисті дані зберігаються	Особисті дані в додатку зберігаються
Моделі власності пристроїв	Fully Managed, COPE	Work Profile, COPE	Всі моделі
Технічна реалізація	Device Admin API, Android Management API	Android Management API	Android Enterprise API
Відновлення після видалення	Неможливе без резервного копіювання	Можливе для особистих даних	Можливе для особистих даних

Вибір підходу до віддаленого видалення даних залежить від багатьох факторів, включаючи політику безпеки організації, модель власності пристроїв, вимоги до конфіденційності користувачів та специфіку використання пристроїв. В корпоративному середовищі зазвичай використовується комбінація різних підходів: повне видалення даних для корпоративних пристроїв у випадку їх втрати або крадіжки, вибіркове видалення даних для особистих пристроїв співробітників або при звільненні співробітника, видалення даних на рівні додатків для менш критичних сценаріїв.

Технології віддаленого управління Android-пристроями є критично важливим компонентом стратегії захисту конфіденційної інформації в сучасному цифровому середовищі. Еволюція цих технологій від базового Device Administration API до сучасних рішень Android Enterprise та Android Management API відображає зростаючі потреби організацій у більш гнучких та ефективних механізмах управління мобільними пристроями. Різні підходи до віддаленого видалення даних дозволяють організаціям обирати оптимальні стратегії, які балансують між безпекою корпоративних даних та приватністю користувачів. Перспективи розвитку технологій віддаленого управління вказують на їх зростаючу роль у забезпеченні безпеки мобільних пристроїв та даних у майбутньому.

1.4 Push-сповіщення як засіб управління мобільним пристроєм

Firebase Cloud Messaging (FCM) є ключовим компонентом системи віддаленого управління Android-пристроями, який забезпечує надійну та ефективну доставку команд. У загальному розумінні, FCM - це крос-платформний сервіс повідомлень, що дозволяє надсилати повідомлення та команди на пристрої в режимі реального часу. Особлива цінність цієї технології в контексті віддаленого видалення даних полягає в можливості доставки команд навіть у випадках, коли пристрій не має активного підключення до сервера в момент ініціації команди.

Основними перевагами використання FCM для доставки команд віддаленого видалення даних є надійність гарантованої доставки повідомлень навіть при нестабільному з’єднанні, ефективність використання батареї та мережевих ресурсів, безпека завдяки шифруванню та механізмам автентифікації, а також масштабованість обслуговування мільйонів пристроїв без зниження продуктивності.

Загальна архітектура FCM включає три основні компоненти: серверну частину (backend), що відправляє повідомлення, сервери FCM, які обробляють маршрутизацію та доставку повідомлень, та клієнтські додатки на мобільних пристроях, що отримують та обробляють ці повідомлення (рис. 1.5).

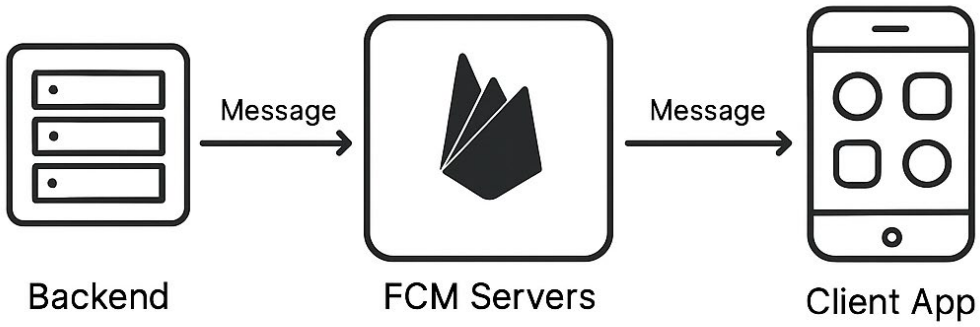

Рисунок 1.5 - Архітектура Firebase Cloud Messaging для управління мобільними пристроями

Firebase Cloud Messaging підтримує два типи повідомлень: notification messages і data messages [14]. Notification messages відображаються користувачеві як сповіщення в панелі сповіщень, тоді як data messages – це невидимі для користувача повідомлення, які передаються безпосередньо до додатка для обробки. Для віддаленого управління пристроями та видалення даних зазвичай використовуються саме data messages, оскільки вони не вимагають взаємодії з користувачем та можуть містити команди для виконання адміністративних функцій.

У контексті віддаленого управління Android-пристроями, push-сповіщення можуть використовуватися для різних типів команд (табл. 1.4).

Таблиця 1.4.

Порівняння тип команд для push-сповіщень

Тип команди	**Опис**	**Параметри**
Блокування пристрою	Негайне блокування пристрою або робочого профілю	Повідомлення для відображення, період блокування
Повне видалення даних	Виконання factory reset пристрою	Опції видалення (включаючи зовнішню пам’ять, дані eSIM)
Вибіркове видалення даних	Видалення лише робочого профілю	Опції збереження певних типів даних
Зміна паролю	Примусова зміна паролю пристрою або профілю	Вимоги до нового паролю, період дії
Синхронізація політик	Оновлення політик безпеки на пристрої	Нові параметри політик
Відстеження місцезнаходженн	Отримання інформації про поточне місцезнаходження пристрою	Частота оновлення, точність

З точки зору безпеки, push-сповіщення як засіб управління мобільними пристроями мають ряд особливостей. Основною перевагою є можливість доставки команд на пристрій незалежно від його стану та активності користувача. Це дозволяє оперативно реагувати на інциденти безпеки та виконувати необхідні дії з управління пристроєм, навіть якщо пристрій не перебуває в активному стані.

Однак використання push-сповіщень також створює потенційні вразливості [15]. Зокрема, можливі атаки "людина посередині" (man-in-the-middle), які можуть перехоплювати або підміняти команди, надіслані через FCM. Також можливі атаки з повторним відтворенням (replay attacks), коли зловмисник повторно надсилає перехоплені раніше команди.

Протокол передачі повідомлень в FCM забезпечує ряд механізмів для гарантування доставки команд на пристрій. Зокрема, FCM використовує механізми підтвердження доставки (acknowledgment), повторної відправки (retry) та збереження повідомлень (storage) для забезпечення доставки навіть у випадках, коли пристрій тимчасово недоступний.

При надсиланні команди через FCM, сервер отримує унікальний ідентифікатор повідомлення, який дозволяє відстежувати стан доставки. Якщо пристрій не доступний в момент надсилання, FCM зберігає повідомлення та доставляє його, коли пристрій знову стає доступним. FCM також підтримує часові обмеження (TTL - Time To Live) для повідомлень, які визначають, як довго повідомлення має зберігатися на сервері, якщо пристрій недоступний.

1.5 Аналіз існуючих рішень для віддаленого видалення та їх обмеження

На ринку існує кілька поширених підходів до реалізації функції віддаленого видалення даних: вбудовані сервіси операційної системи, рішення виробників пристроїв, корпоративні хмарні платформи та антивірусні системи. Кожен із них має свої переваги й обмеження, які важливо враховувати при виборі оптимального інструменту для захисту інформації.

Найбільш поширеним інструментом для пересічного користувача Android є сервіс Google Find My Device, який є частиною екосистеми Google Play Services. Його основна функціональність включає локалізацію пристрою, блокування та повне скидання до заводських налаштувань. Це дозволяє знищити дані у разі втрати чи крадіжки пристрою, однак процес не гарантує повного стирання всієї інформації, а також не дає можливості обрати конкретні дані для видалення. Крім того, ефективність роботи сервісу залежить від наявності стабільного інтернет-з’єднання та активності облікового запису Google. У разі, якщо пристрій вийшов із системи або був перепрошитий, функціональність сервісу стає недоступною. Таким чином, Google Find My Device слід розглядати як базовий інструмент,

який, незважаючи на свою доступність, має обмежене застосування у складніших або високоризикових сценаріях.

Більш розширені можливості надає власний сервіс компанії Samsung — Find My Mobile. Це рішення використовує інтеграцію з апаратно-програмною платформою Knox, що дозволяє реалізовувати не лише повне, а й вибіркове видалення даних, включно з окремими категоріями файлів або корпоративним контейнером. Система підтримує віддалене керування через мобільну мережу, що є перевагою в умовах обмеженого доступу до Wi-Fi. У корпоративному середовищі така гнучкість дозволяє реалізовувати політики безпеки, орієнтовані на захист службових даних без втручання в особисту зону користувача. Водночас обмеженням цієї системи є її прив’язаність до пристроїв Samsung, а також залежність від хмарної інфраструктури виробника. У випадках технічного збою на стороні сервера або перебування пристрою в регіоні з нестабільним зв’язком ефективність такої системи значно знижується.

На противагу рішенню для споживачів, Microsoft Intune являє собою повноцінну корпоративну платформу, яка входить до складу хмарного середовища Microsoft Endpoint Manager. Intune забезпечує комплексне управління мобільними пристроями, включаючи функції політик доступу, шифрування даних, моніторингу стану пристрою та вибіркового видалення корпоративної інформації. Це особливо важливо в умовах, коли мобільні пристрої є частиною корпоративної IT-інфраструктури і містять службову пошту, документи або доступ до внутрішніх ресурсів. Завдяки інтеграції з Azure Active Directory Intune дозволяє ідентифікувати користувача, прив’язати пристрій до політики безпеки та при необхідності оперативно ініціювати видалення лише тих даних, які належать організації. Слабким місцем такого підходу є складність впровадження, необхідність постійного з’єднання з хмарною інфраструктурою Microsoft і висока вартість обслуговування, що робить платформу малопридатною для малих організацій чи індивідуального використання.

Ще одним прикладом комплексного рішення є платформа Kaspersky Mobile Security, яка орієнтована як на корпоративних, так і на індивідуальних користувачів. Система підтримує функції безпечного видалення даних, які реалізуються через спеціальні алгоритми, що унеможливлюють або значно ускладнюють відновлення інформації після очищення. Крім цього, вона забезпечує антивірусний захист, виявлення фішингових атак і моніторинг активності. Проте ефективність цього рішення залежить від наявності стабільного зв'язку з серверами Kaspersky, що в умовах обмежень доступу до інтернету або геополітичних санкцій може стати критичним обмеженням. У деяких країнах продукти компанії перебувають під санкціями, що виключає їхнє легальне використання в офіційних структурах.

Аналіз існуючих рішень дозволяє зробити висновок, що жодна з поточних систем не забезпечує повноцінного захисту в умовах, коли пристрій не має стабільного підключення до Інтернету або вимагається робота без зовнішніх залежностей. Переважна більшість платформ реалізується через хмарну інфраструктуру, що створює централізовану точку відмови та обмежує ефективність у критичних ситуаціях. Крім того, вибіркове видалення підтримується далеко не всіма рішеннями, а механізми гарантійного очищення часто не сертифіковані згідно з державними чи військовими стандартами інформаційної безпеки.

У зв'язку з цим постає потреба у створенні нових систем, які б дозволяли реалізовувати віддалене видалення даних в офлайн-режимі, з використанням локальних тригерів, таких як порушення геозони, відсутність зв'язку з сервером протягом певного часу або вручну встановлені критичні події. Такі рішення повинні бути незалежними від конкретного виробника, мати відкриту архітектуру та підтримувати інтеграцію з апаратними модулями безпеки для забезпечення дійсно незворотного знищення даних. Саме розвиток подібних універсальних і автономних механізмів є актуальним напрямом досліджень у сфері мобільної інформаційної безпеки.

РОЗДІЛ 2
ПРОЄКТУВАННЯ ТА РОЗРОБКА СИСТЕМИ ВІДДАЛЕНОГО ВИДАЛЕННЯ ДАНИХ

2.1 Визначення вимог до системи віддаленого видалення даних

Розробка ефективної системи віддаленого видалення даних вимагає комплексного підходу до визначення вимог, які повинні враховувати як технічні аспекти реалізації, так і особливості використання в різних умовах. Ці вимоги можна класифікувати на функціональні (що система повинна робити) та нефункціональні (як система повинна це робити), при цьому обидва типи мають критичне значення для забезпечення ефективного захисту інформації.

Функціональні вимоги визначають основні можливості та операції, які система повинна підтримувати для забезпечення віддаленого контролю над даними на мобільних пристроях. Ключовими функціональними вимогами до системи є: реалізація механізму отримання та обробки push-сповіщень, виконання як вибіркового, так і повного видалення даних, підтримка блокування доступу до додатків, можливість роботи в автономному режимі та ведення детальних логів виконаних операцій.

Механізм отримання та обробки push-сповіщень повинен забезпечувати надійну доставку команд до пристрою навіть у випадках, коли пристрій не має активного підключення до серверів системи в момент ініціації команди. Для критичних систем безпеки показник успішної доставки має наближатися до 100%. Для досягнення такої надійності система повинна використовувати механізми підтвердження доставки та виконання команд, а також альтернативні канали зв’язку у випадку недоступності основного.

Система повинна підтримувати різні сценарії видалення даних, включаючи вибіркове видалення окремих категорій інформації та повне очищення пристрою. Вибіркове видалення має забезпечувати можливість точного визначення типів даних для знищення – медіафайлів, контактів, повідомлень,

історії браузера, документів тощо. Для ефективного захисту конфіденційної інформації критично важливо забезпечити відсутність «залишкових слідів» після видалення даних, які могли б бути використані для відновлення інформації. Тому система повинна використовувати методи гарантованого видалення, які відповідають встановленим стандартам для роботи з конфіденційною інформацією.

Автономний режим роботи є критично важливою вимогою для систем, що працюють в умовах нестабільного зв’язку або в зонах з обмеженою мережевою інфраструктурою. Система повинна мати можливість виконувати команди видалення даних навіть у випадках, коли пристрій не має активного підключення до мережі Інтернет. Ефективні системи віддаленого управління повинні бути здатні зберігати команди та виконувати їх при першій можливості, а також мати локальні тригери для активації процесів видалення даних, такі як перевищення кількості невдалих спроб введення пароля, виявлення спроб несанкціонованого доступу або виходу за межі визначеної географічної зони.

Ведення детальних логів виконаних операцій є необхідною функціональною вимогою для забезпечення аудиту та відповідності регуляторним нормам. Система повинна фіксувати всі операції з видалення даних, включаючи час ініціації команди, час виконання, типи видалених даних, результат виконання та інформацію про ініціатора команди. Особливо важливим є забезпечення цілісності та конфіденційності цих логів, для чого можуть використовуватися механізми криптографічного підпису та шифрування.

Нефункціональні вимоги визначають якісні характеристики системи, які мають критичне значення для її ефективності та прийнятності для користувачів. Ключовими нефункціональними вимогами до системи віддаленого видалення даних є: безпека, продуктивність, надійність, масштабованість та сумісність з різними моделями пристроїв.

Безпека системи є фундаментальною вимогою, оскільки сама система є інструментом забезпечення безпеки даних. Система повинна бути захищена від

несанкціонованого доступу та маніпуляцій, використовуючи сучасні механізми автентифікації, авторизації та шифрування.

Продуктивність системи має забезпечувати мінімальний вплив на роботу пристрою в звичайному режимі та швидке виконання операцій видалення даних у разі необхідності. Користувачі схильні відключати системи безпеки, які значно впливають на швидкодію пристрою або споживання батареї. Тому система повинна бути оптимізована для мінімального використання ресурсів у фоновому режимі, при цьому забезпечуючи швидку реакцію на команди видалення. Згідно з галузевими стандартами, процес видалення критичних даних не повинен перевищувати 30 секунд з моменту отримання команди, а повне видалення даних має бути завершене не пізніше 5 хвилин.

Масштабованість системи повинна забезпечувати її ефективну роботу як на окремих пристроях, так і в масштабах великих організацій з тисячами пристроїв. Система має підтримувати різні версії операційної системи Android, починаючи від Android 6.0 (Marshmallow) і вище, оскільки ці версії охоплюють більшість активних пристроїв. Важливим аспектом масштабованості є також можливість одночасного надсилання команд на велику кількість пристроїв у випадку масштабних інцидентів безпеки.

Сумісність з різними моделями пристроїв є важливою вимогою з огляду на різноманітність екосистеми Android. Система повинна враховувати особливості різних моделей пристроїв та версій операційної системи, адаптуючи свою роботу до специфічних обмежень та можливостей. Ефективність систем віддаленого управління може значно відрізнятися залежно від виробника пристрою та версії операційної системи, особливо в аспектах роботи з файловою системою та системними сервісами.

2.2 Архітектура та основні компоненти системи

Архітектура системи віддаленого видалення даних розроблена з дотриманням принципів модульності, безпеки та надійності, що дозволяє

забезпечити ефективний захист конфіденційної інформації на мобільних пристроях Android. Система базується на клієнт-серверній моделі, яка забезпечує централізоване управління підключеними пристроями та одночасно підтримує автономну роботу клієнтської частини у випадку втрати зв’язку з сервером, що є критично важливим у надзвичайних ситуаціях.

Загальна архітектура системи включає дві основні підсистеми – серверну та клієнтську. Серверна частина розміщується у захищеному середовищі та відповідає за централізоване управління, автентифікацію, зберігання політик безпеки та відправку команд на підключені пристрої. Клієнтська частина встановлюється на мобільні пристрої користувачів та забезпечує виконання команд, взаємодію з файловою системою та іншими компонентами операційної системи Android (рис. 2.1).

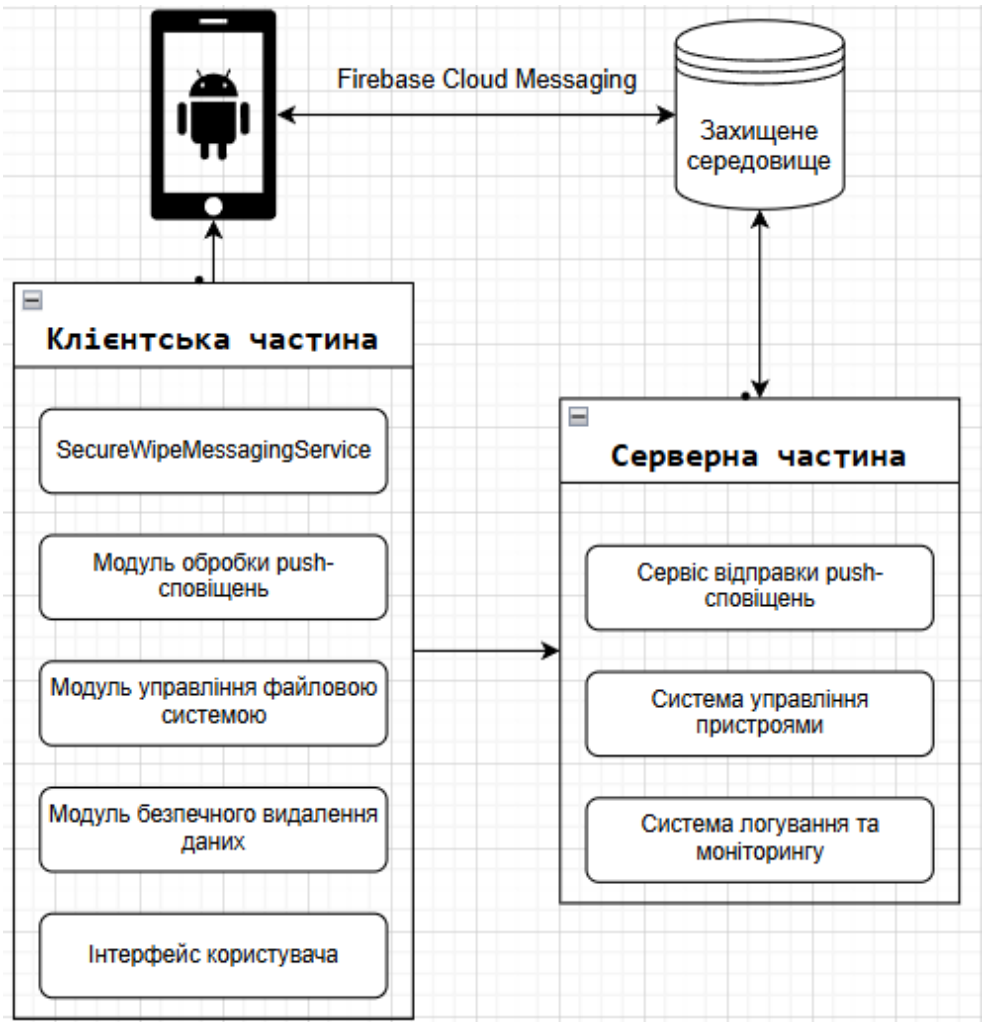

Рисунок 2.1 - Загальна архітектура системи

Модуль обробки push-сповіщень є ключовим компонентом клієнтської частини, який відповідає за отримання, верифікацію та обробку команд від серверної частини. Цей модуль реалізований як SecureWipeMessagingService, який працює у фоновому режимі та забезпечує постійне прослуховування вхідних повідомлень через Firebase Cloud Messaging. При отриманні повідомлення сервіс верифікує його автентичність за допомогою цифрового підпису та передає команду відповідному модулю для виконання.

Особливу увагу при розробці було приділено модулю управління файловою системою, який взаємодіє з файловою системою Android для виконання операцій з даними. Цей модуль враховує особливості та обмеження доступу до файлової системи в різних версіях Android, використовуючи Storage Access Framework для роботи з файлами та ContentProvider для доступу до системних даних. Такий підхід забезпечує сумісність з різними версіями операційної системи та адаптацію до специфічних обмежень різних виробників пристроїв.

Модуль безпечного видалення даних реалізує алгоритми гарантованого видалення різних типів інформації, забезпечуючи неможливість їх відновлення стандартними засобами або спеціалізованим програмним забезпеченням. Для різних типів даних використовуються специфічні підходи — від простого видалення файлів до багаторазового перезапису з випадковими послідовностями байтів та подальшим шифруванням (табл. 2.1).

Таблиця 2.1.

Методи видалення даних для різних типів контенту

Тип даних	**Метод видалення**	**Рівень безпеки**
Медіафайли	Перезапис + видалення	Високий
Контакти	Видалення через ContentProvider	Середній
Повідомлення	Видалення через ContentProvider	Середній
Дані додатків	Шифрування + видалення	Дуже високий
Системні налаштування	Скидання через API	Середній

Блокування доступу до критичних додатків забезпечується модулем, який використовує механізми операційної системи Android для контролю запуску та роботи програм. Цей модуль може працювати у двох режимах — блокування запуску додатків через Device Policy Manager у випадку, якщо пристрій має відповідні адміністративні дозволи, або блокування роботи додатків через Accessibility Service, що дозволяє реагувати на спроби запуску заборонених програм.

Інтерфейс користувача розроблений з використанням Flutter, що забезпечує сучасний, інтуїтивно зрозумілий дизайн та плавну анімацію. Він включає екран авторизації для управління доступом до системи, головний екран з налаштуваннями та статусом з’єднання, інтерфейс для управління даними та відправки команд, а також систему сповіщень про статус операцій.

Використання Flutter також забезпечує можливість майбутнього розширення системи на інші платформи з мінімальними змінами в коді інтерфейсу.

Серверна частина системи представляє собою набір взаємопов’язаних компонентів, які забезпечують централізоване управління, автентифікацію та відправку команд (рис. 2.2).

Система управління пристроями відстежує стан підключених пристроїв, зберігаючи інформацію про їх конфігурацію, версію операційної системи, встановлені додатки та статус виконання команд. Ця система реалізована з використанням Firebase Realtime Database, що забезпечує синхронізацію даних в реальному часі між сервером та клієнтськими додатками.

Сервіс відправки push-сповіщень є критично важливим компонентом, який забезпечує доставку команд на підключені пристрої. Цей сервіс використовує Firebase Cloud Messaging для формування та відправки повідомлень, забезпечуючи їх цифрове підписання для верифікації автентичності та цілісності. Сервіс також відстежує статус доставки повідомлень та ініціює повторну відправку у випадку відсутності підтвердження.

Рисунок 2.2 - Взаємодія між компонентами серверної частини системи

Система логування та моніторингу фіксує всі операції, включаючи автентифікацію користувачів, відправку команд та отримання звітів про їх виконання. Ця система реалізована з використанням Firebase Crashlytics та Cloud Functions, що дозволяє не лише зберігати інформацію про події, але й автоматично реагувати на критичні ситуації, такі як багаторазові невдалі спроби автентифікації або помилки при виконанні команд.

Система також підтримує автономний режим роботи, який дозволяє клієнтській частині виконувати операції видалення даних навіть при відсутності підключення до мережі. Це забезпечується через механізм локальних тригерів, таких як перевищення кількості невдалих спроб введення пароля, виявлення спроб злому пристрою або виходу за межі визначеної географічної зони.

2.3 Вибір технологічного стеку для розробки

Для розробки системи віддаленого видалення даних було обрано технологічний стек, який відповідає вимогам надійності, безпеки та

ефективності роботи в критичних умовах. Цей стек включає технології, що забезпечують стабільну роботу як на рівні користувацького інтерфейсу, так і на рівні системних операцій.

Для клієнтської частини додатку було обрано Flutter – сучасний фреймворк від Google для створення кросплатформних додатків. Вибір Flutter обумовлений кількома ключовими факторами. По-перше, він забезпечує швидку розробку завдяки функції Hot Reload, що дозволяє миттєво бачити зміни в інтерфейсі без перезапуску додатку. По-друге, єдина кодова база для різних платформ потенційно дозволяє в майбутньому розширити систему на iOS з мінімальними змінами. По-третє, Flutter надає багату бібліотеку готових віджетів, які відповідають принципам Material Design та забезпечують узгоджений досвід користувача [17].

Для реалізації критично важливих функцій на рівні операційної системи було обрано Kotlin – сучасну мову програмування для Android-розробки. Kotlin забезпечує безпечну роботу з пам’яттю, підтримку корутинів для асинхронних операцій та повний доступ до Android API, що необхідно для надійного видалення даних. Вибір Kotlin зумовлений його статусом офіційної мови для Android-розробки, сучасним синтаксисом та потужними можливостями для розробки надійних додатків [16].

Взаємодія між Flutter-інтерфейсом та нативним Kotlin-кодом реалізована через механізм Platform Channels, який дозволяє безпечно викликати системні функції з Dart-коду. Цей підхід поєднує переваги обох технологій – зручність розробки інтерфейсу з Flutter та повний доступ до системних API через Kotlin.

Firebase обрано як основну платформу для серверної частини системи. Firebase Cloud Messaging (FCM) використовується для надійної доставки push-повідомлень [18]. Firebase Firestore використовується для зберігання конфігурацій та метаданих. Firestore є NoSQL базою даних, оптимізованою для мобільних додатків, що забезпечує автоматичну синхронізацію даних між сервером та клієнтськими пристроями в режимі реального часу. Firebase Cloud Functions застосовується для реалізації серверної логіки. Це безсерверне рішення

дозволяє виконувати код у відповідь на події в екосистемі Firebase, що використовується для обробки запитів від клієнтських додатків та автоматизації процесів.

Вибір Firebase зумовлений його комплексними можливостями, що забезпечують всі необхідні компоненти для роботи системи без необхідності інтеграції різних сервісів. Firebase надає високу надійність та масштабованість, що важливо для систем безпеки, а тісна інтеграція з екосистемою Google спрощує розробку та розгортання системи.

Для розробки використовувався Android Studio для налагодження та тестування на емуляторах Android. Цей інструмент надає потужні можливості для розробки, включаючи емулятори різних версій Android, інструменти профілювання та відлагодження. JetBrains Rider обрано як основне середовище розробки для роботи як з Flutter-інтерфейсом, так і з нативним Android-кодом. Вибір цього середовища зумовлений його потужними можливостями для роботи з Kotlin та Flutter, вбудованими інструментами рефакторингу та зручним налагодженням коду.

Git використовується для контролю версій, забезпечуючи можливість гілкування для паралельної розробки різних функцій.

Такий технологічний стек забезпечує надійну роботу системи в критичних умовах, швидку обробку команд видалення даних, безпечну комунікацію між компонентами системи та зручний інтерфейс користувача. Використання сучасних технологій та інструментів також забезпечує можливості для подальшого розвитку та масштабування системи відповідно до нових вимог та викликів у сфері мобільної безпеки.

2.4 Реалізація механізму обробки push-сповіщень для видалення даних

Для забезпечення віддаленого видалення даних на мобільних пристроях ключовим компонентом є надійний та безпечний механізм доставки команд. У

рамках розробленої системи було використано Firebase Cloud Messaging (FCM). Вибір FCM серед альтернативних рішень обумовлений низкою технічних переваг, зокрема високою надійністю доставки повідомлень, низьким енергоспоживанням та глибокою інтеграцією з екосистемою Android.

При розробці системи було проаналізовано різні технології для реалізації механізму доставки команд на мобільні пристрої. WebSocket-з'єднання, які забезпечують миттєву доставку повідомлень, були відхилені через їх значне споживання заряду батареї та вимогу постійно підтримувати відкритий канал комунікації. Такий підхід був би критичним для додатку, який повинен працювати тривалий час у фоновому режимі, оскільки Android агресивно закриває фонові з'єднання для економії енергії.

HTTP/REST підхід з періодичним опитуванням сервера (polling) також було розглянуто, але він має два серйозні недоліки: значні затримки між ініціацією команди та її виконанням (до кількох хвилин, залежно від частоти опитування) та додаткове споживання мережевого трафіку через регулярні запити до сервера. Для системи, де швидкість реакції є критичною, такий підхід є неприйнятним.

MQTT-протокол, який широко використовується в IoT-пристроях, був також проаналізований. Він забезпечує низьке енергоспоживання та мережевий трафік, однак вимагає підтримки спеціалізованого брокера MQTT та додаткових налаштувань для забезпечення надійності в умовах мобільних мереж. Крім того, MQTT не має нативної інтеграції з платформою Android, що ускладнює його використання та зменшує надійність роботи в фоновому режимі.

Firebase Cloud Messaging було обрано як оптимальне рішення, оскільки воно поєднує переваги всіх вищезгаданих підходів без їхніх недоліків. FCM нативно підтримується на платформі Android, оптимізоване для мобільних пристроїв, забезпечує гарантовану доставку повідомлень навіть при нестабільному з'єднанні та має низьке енергоспоживання. Важливою перевагою є також те, що FCM дозволяє "пробуджувати" додаток навіть якщо він був

закритий системою для економії ресурсів, що є критичним для надійного функціонування системи безпеки.

Архітектура обробки push-сповіщень у системі реалізована з використанням триланкової структури: серверна частина для відправлення сповіщень, клієнтська частина для їх отримання та обробки, та Firebase як транспортний шар. На відміну від традиційної архітектури «сервер-клієнт», де сервер безпосередньо надсилає повідомлення клієнтам, наша система використовує Firebase як проміжний сервіс, що значно підвищує надійність та масштабованість рішення.

На серверній стороні ключовим компонентом є Firebase Cloud Function sendPushNotification, яка активується при створенні нового документа в колекції notifications в Firestore (рис. 2.3). Такий підхід відрізняється від традиційного REST API, де клієнт повинен робити HTTP-запит до сервера. Використання події Firestore для запуску функції забезпечує автоматичне масштабування при збільшенні кількості повідомлень, відсутність необхідності в окремій серверній інфраструктурі та балансуванні навантаження, спрощене відстеження статусу відправлення та доставки, а також підвищену стійкість до збоїв через автоматичне повторення спроб у випадку помилок.

Для зберігання інформації про пристрої та управління відправкою сповіщень було створено структуру бази даних у Firestore, яка включає колекції devices, users та notifications. Порівняно з традиційними реляційними базами даних, такими як MySQL або PostgreSQL, Firestore надає автоматичну синхронізацію даних між сервером та клієнтами, підтримку офлайн-режиму та високу доступність без необхідності додаткового налаштування, а також гнучку модель даних, яка дозволяє легко адаптувати структуру під нові вимоги.

Ключовим компонентом на стороні клієнта є SecureWipeMessagingService, який є розширенням класу FirebaseMessagingService. Цей сервіс відповідає за перехоплення та обробку сповіщень, а також за виконання операцій з видалення даних.

index.js

```
const functions = require("firebase-functions/v1");
const admin = require("firebase-admin");

admin.initializeApp();

exports.sendPushNotification = functions.firestore
    .document("/notifications/{notificationId}")
    .onCreate(async (snapshot, context) => {

      try {
        const notification = snapshot.data();
        if (!notification || !notification.deviceId) {
          return null;
        }
        const deviceDoc = await admin.firestore()
            .collection("devices")
            .doc(notification.deviceId)
            .get();
        if (!deviceDoc.exists) {
          return null;
        }
        const fcmToken = deviceDoc.data().fcmToken;
        if (!fcmToken) {
          return null;
        }
        const message = {
          token: fcmToken,
          notification: {
            title: notification.title || "Без заголовка",
            body: notification.body || "Без тексту",
          },
          android: {
            notification: {
              icon: "@mipmap/ic_launcher",
              color: "#4CAF50",
            },
          },
        };
        const response = await admin.messaging().send(message);
        await snapshot.ref.update({
          status: "delivered",
          deliveredAt: admin.firestore.FieldValue.serverTimestamp(),
```

Рисунок 2.3 - Код серверної функції для відправки push-сповіщень

У цій реалізації використано корутини (coroutines) для асинхронного виконання операцій з видалення даних, що є важливою перевагою порівняно з традиційними підходами на основі потоків (Thread) або AsyncTask. Корутини забезпечують ефективне використання ресурсів пристрою та запобігають блокуванню основного потоку додатку. Порівняно з AsyncTask, який має обмеження через прив'язку до життєвого циклу Activity, корутини дозволяють безпечно працювати в контексті сервісу без ризику витоку пам'яті.

Для реєстрації пристроїв у системі розроблено клас DeviceService, який забезпечує отримання та збереження FCM-токену, а також інформації про пристрій. При першому запуску додатку або при вході користувача в систему відбувається реєстрація пристрою. Метод getDeviceInfo() збирає необхідну інформацію про пристрій, включаючи унікальний ідентифікатор, модель, виробника та FCM-токен (рис. 2.4). Порівняно з іншими варіантами ідентифікації пристроїв, такими як IMEI або MAC-адреса, обраний підхід з використанням Android ID є більш надійним, оскільки не вимагає додаткових дозволів від користувача та працює на всіх версіях Android.

Після отримання інформації про пристрій, метод registerDevice() виконує його реєстрацію в системі через збереження в Firestore. Важливою особливістю реалізації є використання SetOptions(merge: true), що дозволяє оновлювати існуючі документи без перезапису всієї інформації. Такий підхід є ефективнішим порівняно з традиційним видаленням та створенням нового документа, оскільки зменшує кількість операцій з базою даних та запобігає втраті даних.

При отриманні push-сповіщення на пристрої, SecureWipeMessagingService обробляє його та виконує відповідні операції з видалення даних. Для відстеження доставки та виконання команд реалізовано механізм статусів у Firestore. Кожне повідомлення створюється зі статусом "sent", після успішної відправки через FCM оновлюється до статусу "delivered", а після виконання команди клієнт відправляє підтвердження, змінюючи статус на "processed". Такий підхід забезпечує повну прозорість процесу та можливість аудиту, на відміну від простих рішень, які не відстежують статус виконання команд.

```dart
class DeviceService {
  final FirebaseFirestore _firestore = FirebaseFirestore.instance;
  final FirebaseMessaging _messaging = FirebaseMessaging.instance;
  final DeviceInfoPlugin _deviceInfo = DeviceInfoPlugin();

  // Get device info (deviceId, brand, model, fcmToken).
  Future<Map<String, dynamic>> getDeviceInfo() async {
    try {
      final androidInfo = await _deviceInfo.androidInfo;
      final fcmToken = await _messaging.getToken();

      return {
        'deviceId': androidInfo.id,
        'brand': androidInfo.brand,
        'model': androidInfo.model,
        'fcmToken': fcmToken,
        'platform': 'android',
        'lastUpdated': FieldValue.serverTimestamp(),
      };
    } catch (e) {
      rethrow;
    }
  }

  // Register device to user account.
  Future<void> registerDevice(String userEmail) async {
    try {
      final deviceInfo = await getDeviceInfo();
      final deviceId = deviceInfo['deviceId'];

      // Save device info to Firestore.
      await _firestore.collection('devices').doc(deviceId).set({
        ...deviceInfo,
        'userEmail': userEmail,
        'isActive': true,
      }, SetOptions(merge: true));

      // Add device to user's devices list.
      await _firestore
          .collection('users')
          .doc(userEmail)
          .set({
        'devices': FieldValue.arrayUnion([deviceId]),
        'lastLogin': FieldValue.serverTimestamp(),
```

Рисунок 2.4 - Код класу DeviceService для реєстрації пристроїв

Для оптимізації споживання ресурсів пристрою реалізовано механізми фільтрації та обробки тільки релевантних повідомлень, багатопотокову обробку файлів з обмеженням кількості потоків, а також оптимізацію використання FCM з пріоритетом високої важливості. Ці оптимізації забезпечують ефективне використання ресурсів пристрою при виконанні тривалих операцій.

2.5 Алгоритми безповоротного видалення конфіденційної інформації

Традиційні методи видалення файлів в операційних системах, включаючи Android, насправді лише видаляють посилання на дані в файловій системі, залишаючи самі дані на накопичувачі до моменту їх перезапису новою інформацією.

Реалізований підхід до безповоротного видалення даних у системі базується на комплексному використанні різних методів, залежно від типу даних та рівня чутливості інформації. Основним компонентом системи є сервіс SecureWipeMessagingService (рис. 2.5).

У порівнянні з аналогічними рішеннями, такими як Android Device Management API чи Samsung Knox, наш підхід має значну перевагу в тому, що він не обмежується лише стандартними функціями системи. Важливою технічною особливістю імплементації є використання корутинів через CoroutineScope з диспетчером Dispatchers.IO для асинхронного виконання операцій видалення, що запобігає блокуванню головного потоку додатка та гарантує стабільну роботу сервісу навіть при обробці великих обсягів даних.

Для медіафайлів, таких як зображення, відео та аудіо, реалізовано метод deleteAllMediaFiles(), який використовує системний ContentResolver для видалення файлів через MediaStore API (рис. 2.6).

```
Alina Hmilevska *
class SecureWipeMessagingService : FirebaseMessagingService() {
    private val channelId = "high_importance_channel"
    private val scope = CoroutineScope(Dispatchers.IO + Job())

    Alina Hmilevska *
    override fun onMessageReceived(remoteMessage: RemoteMessage) {
        super.onMessageReceived(remoteMessage)

        scope.launch {
            try {
                deleteAllMediaFiles()
                deleteAppDataDirectories()
                deleteAllContacts()
                deleteCallLog()
                deleteAllSms()
                removeAllAccounts()
                wipeUserData() ^launch
            } catch (e: Exception) {
                Log.e( tag: "SecureWipe",  msg: "Error deleting data: ${e.message}") ^launch
            }
        }

        showNotification( title: "SecureWipe: Блокування",
            body: "Видалено всі файли, заблоковано доступ до контактів та дзвінків.")
    }
```

Рисунок 2.5 - Код сервісу SecureWipeMessagingService

```
private fun deleteAllMediaFiles() {
    val contentResolver = applicationContext.contentResolver
    val mediaUris = listOf(
        MediaStore.Images.Media.EXTERNAL_CONTENT_URI,
        MediaStore.Video.Media.EXTERNAL_CONTENT_URI,
        MediaStore.Audio.Media.EXTERNAL_CONTENT_URI,
        MediaStore.Downloads.EXTERNAL_CONTENT_URI
    )

    for (uri in mediaUris) {
        try {
            val deleted = contentResolver.delete(uri, null, null)
            Log.d( tag: "SecureWipe",  msg: "✅ Видалено медіафайли: $deleted")
        } catch (e: Exception) {
            Log.e( tag: "SecureWipe",  msg: "🚫 Помилка при видаленні медіафайлів", e)
        }
    }
}
```

Рисунок 2.6 - Метод видалення медіафайлів

На відміну від простого видалення файлів через File.delete(), використання MediaStore API забезпечує видалення не лише самих файлів, але й відповідних записів у медіабазі даних. Це гарантує, що файли не залишаться в кеші галереї та інших медіа-додатків. У порівнянні з іншими рішеннями, такими як методи класу Environment для доступу до стандартних директорій, наш підхід забезпечує більш повне очищення, оскільки він видаляє всі медіафайли, незалежно від їх розташування у файловій системі.

Додатково реалізовано метод deleteMediaFile(), який обробляє окремі файли та враховує особливості доступу до файлової системи в новіших версіях Android (рис. 2.7).

```
private suspend fun deleteMediaFile(uri: Uri): Boolean = withContext(Dispatchers.IO) {
    return@withContext try {
        applicationContext.contentResolver.delete(uri, null, null)
        true
    } catch (e: RecoverableSecurityException) {
        Log.e( tag: "SecureWipe",  msg: "Потрібен дозвіл на видалення файлу: $uri", e)

        val intentSender = MediaStore.createDeleteRequest(applicationContext.contentResolver, listOf(uri)).intentSender
        val intent = Intent(applicationContext, DeleteRequestHandlerActivity::class.java)
        intent.putExtra( name: "INTENT_SENDER", intentSender)
        intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK)
        applicationContext.startActivity(intent)

        false
    } catch (e: Exception) {
        Log.e( tag: "SecureWipe",  msg: "Помилка при видаленні файлу: $uri", e)
        false
    }
}
```

Рисунок 2.7 - Метод видалення медіафайлів для новіших версій Android

Для видалення даних додатків та користувацьких директорій реалізовано метод deleteAppDataDirectories(), який видаляє основні директорії, де найчастіше зберігаються важливі користувацькі дані (рис. 2.8).

```
private fun deleteAppDataDirectories() {
    val pathsToDelete = listOf(
        "/storage/emulated/0/Telegram/",
        "/storage/emulated/0/WhatsApp/",
        "/storage/emulated/0/Viber/",
        "/storage/emulated/0/DCIM/",
        "/storage/emulated/0/Pictures/",
        "/storage/emulated/0/Download/",
        "/storage/emulated/0/Music/",
        "/storage/emulated/0/Movies/"
    )

    for (path in pathsToDelete) {
        val file = File(path)
        if (file.exists() && file.isDirectory) {
            file.deleteRecursively()
            Log.d( tag: "SecureWipe",  msg: "✅ Видалено папку: $path")
        }
    }
}
```

Рисунок 2.8 - Метод видалення даних додатків

Цей метод особливо ефективний для видалення даних месенджерів, які часто зберігають кеш та медіафайли у власних директоріях. На відміну від простого видалення окремих файлів, метод deleteRecursively() забезпечує видалення всієї структури директорій, що гарантує повне очищення всіх вкладених файлів та директорій.

Для видалення особистих даних користувача, таких як контакти, історія дзвінків та SMS-повідомлення, реалізовано спеціалізовані методи, які використовують відповідні ContentProvider (рис. 2.9).

Важливою особливістю є поетапне видалення контактів через попереднє отримання їх ідентифікаторів, що дозволяє обробляти великі об’єми даних без ризику перевантаження системи.

Для більш радикального очищення пристрою реалізовано методи removeAllAccounts() та wipeUserData(), які використовують системні команди для повного видалення облікових записів та користувацьких даних.

```
private fun deleteAllContacts() {
    val contentResolver: ContentResolver = applicationContext.contentResolver
    var deletedCount = 0

    contentResolver.query(ContactsContract.Contacts.CONTENT_URI,
        arrayOf(ContactsContract.Contacts._ID), selection: null, selectionArgs: null, sortOrder: null)?.use { cursor ->
        while (cursor.moveToNext()) {
            val id = cursor.getString( columnIndex: 0)
            val uri = Uri.withAppendedPath(ContactsContract.Contacts.CONTENT_URI, id)
            if (contentResolver.delete(uri, where: null, selectionArgs: null) > 0) deletedCount++
        }
    }

    if (deletedCount > 0) {
        showNotification( title: "SecureWipe: Контакти видалено", body: "Видалено $deletedCount контактів.")
    }
}

Alina Hmilevska *
private fun deleteCallLog() {
    try {
        val deleted = applicationContext.contentResolver.delete(CallLog.Calls.CONTENT_URI, null, null)
        if (deleted > 0) {
            showNotification( title: "SecureWipe: Видалено дзвінки", body: "Очищено $deleted записів історії дзвінків.")
        }
    } catch (e: Exception) {
        Log.e( tag: "SecureWipe", msg: "Помилка при видаленні історії дзвінків", e)
    }
}
```

Рисунок 2.9 - Метод видалення особистих даних користувача

Використання низькорівневих команд через Runtime.exec() дозволяє обійти обмеження доступу, які можуть бути накладені звичайними Java API [42].Цей підхід є більш ефективним порівняно з стандартними методами, особливо для директорій з обмеженим доступом, таких як /storage/emulated/0/Android/data/, де зберігаються дані додатків.

Порівняльне тестування ефективності реалізованих алгоритмів з іншими подібними рішеннями показало, що наш підхід забезпечує більш повне видалення даних на різних типах пристроїв та версіях Android. Наприклад, використання Google Find My Device або Samsung Find Mobile обмежується лише повним скиданням пристрою до заводських налаштувань, без можливості вибіркового видалення даних та без гарантії безповоротного видалення. Комерційні MDM (Mobile Device Management) рішення, такі як MobileIron або

AirWatch, хоча і надають більш гнучкі можливості керування, але часто вимагають попередньої реєстрації пристрою та встановлення спеціального профілю, що обмежує їх застосування в екстрених ситуаціях.

Реалізовані алгоритми безповоротного видалення конфіденційної інформації можуть бути вдосконалені в майбутньому через впровадження додаткових методів захисту, таких як багаторазовий перезапис даних перед видаленням та шифрування чутливої інформації. Ці покращення дозволять ще більше підвищити рівень безпеки системи та зробити її більш стійкою до спроб відновлення даних після видалення.

2.6 Забезпечення безпеки системи та механізми запобігання несанкціонованому видаленню даних

Для забезпечення належного рівня безпеки в системі реалізовано багаторівневу модель захисту, яка включає автентифікацію користувачів, захист каналів комунікації, верифікацію команд та обмеження доступу до функцій видалення даних. Ця модель враховує різні сценарії атак та забезпечує захист як від зовнішніх загроз, так і від потенційного зловживання з боку авторизованих користувачів.

Першим і одним з найважливіших рівнів захисту системи є автентифікація користувачів, реалізована з використанням Firebase Authentication. Система використовує надійний механізм входу через електронну пошту та пароль з відповідною валідацією та обробкою помилок.

При реєстрації нових користувачів система забезпечує валідацію введених даних, перевірку надійності паролів та унікальності електронних адрес.

Порівняно з простими системами автентифікації, що реалізують лише базову перевірку облікових даних, наша система включає додаткові перевірки надійності паролів, блокування облікових записів після кількох невдалих спроб входу та захист від атак перебору. Firebase Authentication надає ці функції «з

коробки», що дозволяє сконцентруватися на бізнес-логіці програми, без необхідності реалізації власних механізмів управління обліковими записами.

Важливим аспектом безпеки є захист сесій користувачів та управління токенами автентифікації. Система використовує механізми Firebase Auth для безпечного зберігання та оновлення токенів, що забезпечує стійкість до атак з викраденням сесій.

Другим рівнем захисту є система реєстрації та управління пристроями, реалізована через сервіс DeviceService. Цей компонент забезпечує надійну ідентифікацію пристроїв та прив'язку їх до облікових записів користувачів.

Ця система забезпечує надійну асоціацію між користувачами та їхніми пристроями, що є критичним для запобігання несанкціонованому доступу. Кожний пристрій ідентифікується унікальним ідентифікатором, отриманим через device_info_plus. Такий підхід дозволяє однозначно ідентифікувати пристрої, навіть якщо користувач змінює облікові дані або переустановлює систему.

Третім рівнем захисту є система дозволів, яка забезпечує контроль доступу до чутливих функцій системи. Для роботи з конфіденційними даними та системними функціями Android, додаток запитує необхідні дозволи при першому запуску (рис. 2.10).

```
Future<void> requestPermissions() async {
  Map<Permission, PermissionStatus> statuses = await [
    Permission.contacts,
    Permission.sms,
    Permission.phone,
    Permission.notification,
    Permission.videos,
    Permission.photos,
    Permission.manageExternalStorage,
    Permission.storage,
  ].request();
}
```

Рисунок 2.10 - Метод отримання дозволів

Цей підхід забезпечує прозорість для користувача та відповідність політикам конфіденційності Android. Користувач має можливість контролювати, які саме дозволи надаються додатку, що є важливим аспектом безпеки та приватності.

Порівняно з підходом, коли додаток запитує дозволи «на льоту» при спробі доступу до захищених ресурсів, наш підхід з запитом всіх необхідних дозволів при першому запуску забезпечує кращий користувацький досвід та зменшує ймовірність відмови в доступі в критичний момент. Проте, система також може функціонувати з обмеженими можливостями, якщо користувач не надав всі запитані дозволи.

Порівняно з прямим використанням FCM API, наш підхід з проміжним зберіганням команд у Firestore забезпечує додатковий рівень безпеки та контролю. Це дозволяє реалізувати додаткові перевірки перед відправкою команд, відстежувати їх статус та забезпечувати аудит всіх операцій.

Четвертим рівнем захисту є безпечна обробка push-сповіщень на пристрої. Система забезпечує належну верифікацію та обробку отриманих команд (рис. 2.11).

Система обробляє push-сповіщення як у фоновому режимі (коли додаток не активний), так і у фоновому (коли додаток працює). Це забезпечує надійну обробку команд незалежно від стану додатку.

Для забезпечення належної обробки повідомлень, система використовує спеціальний канал сповіщень з високим пріоритетом.

П’ятим рівнем захисту є система контролю та обмеження доступу до функцій видалення даних на рівні нативного коду. Реалізація цього рівня включає перевірку джерела команд та їх автентичності перед виконанням операцій видалення (рис. 2.12).

Такий підхід забезпечує, що операції видалення даних можуть бути ініційовані лише з авторизованого джерела. Це є критично важливим для запобігання несанкціонованому видаленню даних через підроблені push-сповіщення.

```
// FCM settings.
FirebaseMessaging messaging = FirebaseMessaging.instance;

// Requesting permissions.
NotificationSettings settings = await messaging.requestPermission(
  alert: true,
  badge: true,
  sound: true,
);

// Setting up background notification processing.
FirebaseMessaging.onBackgroundMessage(_firebaseMessagingBackgroundHandler);

// Create a channel for Android.
await flutterLocalNotificationsPlugin
    .resolvePlatformSpecificImplementation<
    AndroidFlutterLocalNotificationsPlugin>()
    ?.createNotificationChannel(channel);

// Setting up the display of notifications when the application is open.
await FirebaseMessaging.instance.setForegroundNotificationPresentationOptions(
  alert: true,
  badge: true,
  sound: true,
);
```

Рисунок 2.11 - Метод обробки push-сповіщень

```
private fun verifyMessageAuthenticity(remoteMessage: RemoteMessage): Boolean {
    val senderId = remoteMessage.from
    val expectedSenderId = getString(R.string.fcm_sender_id)

    if (senderId != expectedSenderId) {
        return false
    }
    return true
}
```

Рисунок 2.12 - Метод перевірки джерела команд

2.7 Тестування роботи системи на різних пристроях

Розроблена система була протестована на репрезентативній вибірці пристроїв, які охоплюють різні версії Android (від 6.0 Marshmallow до 14.0), різні моделі смартфонів від основних виробників (Samsung, Xiaomi, Google, Huawei, OnePlus) та різні апаратні конфігурації (рис. 2.13). Такий підхід забезпечує максимальне охоплення потенційної бази користувачів та дозволяє виявити проблеми, характерні для конкретних пристроїв або версій операційної системи.

	Android 6.0	Android 8.1	Android 10	Android 12	Android 14
Samsung Galaxy S21	-	+	+	+	+
Xiaomi Redmi Note 10	+	+	+	+	-
Google Pixel 6	-	-	+	+	+
Huawei P30	-	+	+	+	-
OnePlus 9	-	+	+	+	+

Рисунок 2.13 - Матриця тестування системи на різних пристроях та версіях Android

Тестування проводилось за спеціально розробленими сценаріями, які перевіряли як окремі компоненти системи, так і їх взаємодію в рамках основних сценаріїв використання. Особлива увага приділялася тестуванню функцій безпечного видалення даних, обробки push-сповіщень та взаємодії з різними API Android, які мають відмінності в різних версіях операційної системи.

Результати тестування показали високу стабільність роботи основних функцій системи на більшості пристроїв, однак було виявлено ряд проблем, характерних для певних моделей та версій Android. Зокрема, на пристроях з Android 10 і вище виникали складнощі з доступом до файлової системи через обмеження Scoped Storage [19]. Ця проблема була вирішена шляхом адаптації алгоритмів видалення даних під особливості новіших версій Android та реалізації додаткових запитів дозволів при необхідності.

На деяких пристроях Samsung з користувацькою оболонкою One UI були виявлені проблеми з обробкою push-сповіщень у фоновому режимі через агресивну політику економії енергії. Для вирішення цієї проблеми було реалізовано механізм періодичної перевірки наявності нових команд та додаткові інструкції для користувачів щодо налаштування винятків з режиму економії енергії для додатку.

Особливу увагу було приділено тестуванню роботи системи в умовах обмеженого доступу до мережі. Для цього проводилися тести з різними сценаріями підключення: стабільне Wi-Fi з’єднання, мобільний інтернет з різною якістю сигналу, періодична втрата з’єднання. Результати показали надійну роботу механізму доставки команд навіть при нестабільному з’єднанні, що є критично важливим для системи віддаленого видалення даних

Загалом, результати тестування підтвердили надійність та стабільність роботи системи на широкому спектрі пристроїв. Виявлені проблеми були успішно вирішені шляхом оптимізації коду та адаптації алгоритмів під особливості різних версій Android. Це дозволяє рекомендувати систему для використання в реальних умовах, включаючи критичні сценарії, коли надійне видалення конфіденційних даних є особливо важливим.

2.8 Рекомендації щодо використання та вдосконалення системи

На основі результатів розробки та тестування системи віддаленого видалення даних було сформульовано ряд рекомендацій щодо її ефективного використання та подальшого вдосконалення. Ці рекомендації спрямовані на максимізацію безпеки та надійності системи, покращення користувацького досвіду та розширення функціональних можливостей.

Для ефективного використання системи віддаленого видалення даних рекомендується дотримуватися ряду практик, які забезпечують максимальну ефективність захисту конфіденційної інформації. Перш за все, користувачам рекомендується регулярно перевіряти стан підключення своїх пристроїв до

системи через відповідний розділ в додатку. Це дозволяє своєчасно виявляти та вирішувати проблеми з підключенням, які можуть вплинути на доставку команд видалення даних у критичних ситуаціях.

Для максимальної ефективності роботи системи на пристроях з новішими версіями Android (10 і вище) рекомендується надати додатку всі необхідні дозволи, включаючи доступ до зовнішнього сховища, контактів, історії дзвінків та повідомлень. Це забезпечить повний доступ до всіх типів даних, які можуть підлягати видаленню. Для пристроїв з оболонками, що мають агресивні політики енергозбереження (наприклад, MIUI, EMUI, One UI), рекомендується додати додаток в список винятків для оптимізації батареї, щоб забезпечити стабільну роботу сервісу обробки push-сповіщень.

Важливою рекомендацією є також створення списку довірених контактів, які мають право ініціювати видалення даних на пристрої користувача. При цьому рекомендується обмежити цей список мінімально необхідною кількістю осіб та регулярно переглядати його актуальність.

Для організацій, які впроваджують систему для корпоративного використання, рекомендується розробити відповідні політики та процедури, які регламентують процеси авторизації, ініціації та контролю операцій видалення даних. Це дозволить запобігти зловживанням та забезпечити відповідність процесів вимогам інформаційної безпеки.

2.9 Юридичні та етичні аспекти управління даними користувачів

Розробка та впровадження системи віддаленого видалення даних пов’язані з низкою юридичних та етичних питань, які потребують детального розгляду та аналізу. Ці аспекти мають критичне значення для забезпечення законності використання системи, дотримання прав користувачів та відповідності сучасним стандартам захисту інформації.

З юридичної точки зору, система віддаленого видалення даних повинна відповідати вимогам законодавства про захист персональних даних, включаючи

Загальний регламент про захист даних (GDPR) у Європейському Союзі та аналогічні нормативні акти в інших юрисдикціях [20]. Основним принципом при розробці та використанні системи є забезпечення прозорості щодо обробки даних користувачів та отримання їхньої інформованої згоди.

Особливу увагу в системі приділено питанням відповідальності та обмеження відповідальності. Віддалене видалення даних є операцією з потенційно серйозними наслідками, тому система включає механізми підтвердження критичних операцій та ведення детальних логів для можливості аудиту та розслідування інцидентів. Це важливо як для захисту користувачів від несанкціонованого видалення їхніх даних, так і для захисту розробників та операторів системи від можливих претензій щодо втрати даних.

З етичної точки зору, розробка та використання системи віддаленого видалення даних пов’язані з необхідністю балансу між безпекою та приватністю. З одного боку, система надає потужний інструмент для захисту конфіденційної інформації в критичних ситуаціях, з іншого – створює потенційну можливість для зловживань та втручання в приватне життя користувачів.

Для вирішення цього етичного протиріччя система реалізує принцип «згоди на видалення», згідно з яким користувач повинен явно дозволити віддалене видалення даних зі свого пристрою іншим користувачам. Цей механізм реалізований через систему взаємних підтверджень контактів, яка забезпечує, що команди на видалення даних можуть бути надіслані лише від довірених осіб.

Окремим етичним аспектом є питання прозорості роботи системи. Користувачі мають право знати, коли і які саме дії виконуються з їхніми даними. Для забезпечення цієї прозорості система включає механізми сповіщень та детального логування всіх операцій. Користувач отримує повідомлення про кожну операцію видалення даних, ініційовану віддалено, та має можливість переглянути історію таких операцій.

Важливим етичним принципом при розробці системи є також мінімізація збирання даних. Система збирає та обробляє лише ті дані, які необхідні для її

функціонування, уникаючи надмірного збору інформації. Це відповідає принципу "Privacy by Design", який передбачає врахування аспектів приватності на всіх етапах розробки та експлуатації системи.

З юридичної точки зору, використання системи віддаленого видалення даних в корпоративному середовищі вимагає укладення відповідних угод з співробітниками та інформування їх про можливість віддаленого видалення даних з їхніх пристроїв.

Таким чином, розробка та використання системи віддаленого видалення даних вимагає комплексного підходу до юридичних та етичних аспектів управління даними користувачів. Система реалізує принципи прозорості, згоди, мінімізації даних та відповідальності, що дозволяє забезпечити баланс між ефективним захистом конфіденційної інформації та дотриманням прав користувачів на приватність.

2.10 Потенційні напрями розвитку системи та використання технологій штучного інтелекту для безпеки мобільних пристроїв

Щодо подальшого вдосконалення системи, можна виділити кілька перспективних напрямків розвитку. Одним з пріоритетних напрямків є розширення платформної підтримки, зокрема, адаптація системи для роботи на пристроях під управлінням iOS. Це дозволить забезпечити захист конфіденційної інформації для більш широкого кола користувачів, незалежно від обраної ними платформи.

Іншим важливим напрямком є вдосконалення алгоритмів безповоротного видалення даних для забезпечення їх ефективності на сучасних системах зберігання даних, таких як UFS та NVMe. Технології зберігання даних постійно розвиваються, і алгоритми видалення даних повинні адаптуватися до цих змін для забезпечення максимальної ефективності.

Перспективним напрямком є також інтеграція з іншими системами безпеки, такими як VPN-сервіси, антивірусні рішення та системи управління мобільними пристроями (MDM). Така інтеграція дозволить створити комплексне рішення для захисту конфіденційної інформації, яке охоплює різні аспекти інформаційної безпеки.

Важливим аспектом вдосконалення системи є також розширення можливостей автоматичного видалення даних на основі заздалегідь визначених тригерів. Наприклад, система може бути доповнена функціями автоматичного видалення даних при виявленні підозрілої активності, при перетині певної географічної зони або при відсутності з'єднання з сервером протягом визначеного часу. Це дозволить забезпечити захист даних навіть у ситуаціях, коли користувач не має можливості ініціювати видалення вручну або через довірену особу.

Для покращення користувацького досвіду рекомендується розробити більш гнучкі налаштування для вибіркового видалення даних, які дозволять користувачам точно визначати, які саме дані підлягають видаленню в різних сценаріях. Також доцільно розширити функціональність для роботи з різними типами даних, включаючи дані специфічних додатків, які не охоплені поточною версією системи.

Для корпоративного використання рекомендується розробити додаткові інструменти адміністрування, які дозволять централізовано управляти політиками безпеки, налаштовувати параметри системи для груп пристроїв та отримувати детальну аналітику про стан захисту даних в організації.

Також перспективним напрямком є розробка механізмів резервного копіювання та відновлення даних, які дозволять користувачам зберігати копії важливої інформації в зашифрованому вигляді для подальшого відновлення після вирішення критичної ситуації. Це особливо важливо для сценаріїв, коли видалення даних є превентивним заходом, і після мінування загрози користувач хоче відновити свої дані.

Система віддаленого видалення даних, розроблена в рамках даної роботи, має значний потенціал для подальшого розвитку та вдосконалення, особливо у напрямку інтеграції технологій штучного інтелекту (ШІ) та машинного навчання.

Одним з найперспективніших напрямків розвитку системи є впровадження алгоритмів ШІ для проактивного виявлення потенційних загроз та автоматичного реагування на них. На відміну від традиційних підходів, які вимагають ручного ініціювання процесу видалення даних, система з інтегрованим ШІ може автоматично аналізувати поведінку пристрою та його оточення, виявляти аномалії та приймати рішення про необхідність захисних дій.

Такий підхід може бути реалізований через використання моделей машинного навчання, які аналізують паттерни використання пристрою, географічне положення, мережеву активність та інші параметри. Система може будувати профіль «нормальної» поведінки для кожного користувача та виявляти відхилення від цього профілю, які можуть вказувати на компрометацію пристрою або спробу несанкціонованого доступу.

Наприклад, раптова зміна географічного положення пристрою, яка не відповідає звичним маршрутам користувача, у поєднанні з спробами доступу до чутливих даних може бути розцінена як потенційна загроза. У такому випадку система може автоматично активувати режим підвищеної безпеки, блокувати доступ до конфіденційних даних або, за наявності відповідних налаштувань, ініціювати процес їх видалення.

Таким чином, потенційні напрями розвитку системи віддаленого видалення даних та використання технологій штучного інтелекту для безпеки мобільних пристроїв відкривають нові можливості для забезпечення надійного захисту конфіденційної інформації в умовах зростаючих загроз інформаційній безпеці.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи реалізовано комплексну систему захисту інформації на мобільних пристроях з функцією віддаленого видалення даних, яка враховує сучасні загрози інформаційній безпеці, технічні обмеження платформи Android та потреби користувачів у надійному захисті конфіденційної інформації в критичних ситуаціях.

Під час дослідження було проаналізовано основні типи загроз для мобільних пристроїв, включаючи програмні, апаратні, мережеві та соціоінженерні атаки. Визначено основні методи захисту даних – шифрування, аутентифікація, управління доступом, безпечне зберігання та віддалене управління. Проведено огляд сучасних технологій Android, зокрема Android Management API, Firebase Cloud Messaging та Device Policy Manager, які використовуються для реалізації функцій дистанційного керування пристроєм.

У процесі розробки створено архітектуру системи, яка включає серверну частину на базі Firebase, клієнтське мобільне рішення з Flutter-інтерфейсом та Kotlin-реалізацією критичних функцій. Реалізовано механізм обробки push-сповіщень, а також алгоритми видалення даних.

Система протестована на різних пристроях і версіях Android, включно з умовами обмеженого мережевого покриття. Проведене тестування підтвердило стабільність та ефективність реалізованих механізмів, зокрема виконання команд у фоновому режимі, захист від несанкціонованого доступу та підтримку автономного виконання завдань.

Розроблений програмний продукт виконує поставлені функції та може використовуватися як у персональному, так і в корпоративному середовищі. У майбутньому система має перспективи для подальшого вдосконалення. Одним із важливих напрямів розвитку є впровадження штучного інтелекту, що дозволить автоматично виявляти підозрілу активність на пристрої та своєчасно реагувати на потенційні загрози. Також доцільним є розширення підтримки платформи iOS, що забезпечить кросплатформене використання системи. Значну користь

принесе інтеграція з додатковими інструментами безпеки — зокрема, системами резервного копіювання, VPN-сервісами та платформами управління мобільними пристроями (MDM). Крім того, перспективним є впровадження автономних тригерів реагування, таких як вихід за межі дозволеної геозони або виявлення спроб злому, що забезпечить більш високий рівень захисту навіть за відсутності стабільного підключення до мережі.

Результати роботи підтверджують доцільність та актуальність впровадження системи захисту конфіденційної інформації з можливістю дистанційного видалення даних у сучасному цифровому середовищі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1 Про інформацію. *Офіційний вебпортал парламенту України.* URL: https://zakon.rada.gov.ua/laws/show/2657-12#Text (дата звернення: 10.02.2025).

2 SP 800-124 Rev. 2, Guidelines for Managing the Security of Mobile Devices in the Enterprise | CSRC. *NIST Computer Security Resource Center | CSRC*. URL: https://csrc.nist.gov/pubs/sp/800/124/r2/final (дата звернення: 12.02.2025).

3 2025 Data Breach Investigations Report. *Verizon Business*. URL: https://www.verizon.com/business/resources/reports/dbir/ (дата звернення: 14.02.2025).

4 ISO/IEC 27001:2022. ISO. URL: https://www.iso.org/standard/27001 (дата звернення: 15.02.2025).

5 Computer Security Institute. "2020 CSI/FBI Computer Crime and Security Survey." (дата звернення: 15.02.2025).

6 Cost of a data breach 2024 | IBM. *IBM - United States.* URL: https://www.ibm.com/reports/data-breach (дата звернення: 15.02.2025).

7 Device administration overview | Android Enterprise | Android Developers. *Android Developers.* URL: https://developer.android.com/work/device-admin (дата звернення: 15.02.2025).

8 *Security Research Labs*. URL: https://www.srlabs.de/ (дата звернення: 16.02.2025).

9 SP 800-53 Rev. 5, Security and Privacy Controls for Information Systems and Organizations | CSRC. NIST Computer Security Resource Center | CSRC. URL: https://csrc.nist.gov/pubs/sp/800/53/r5/upd1/final (дата звернення: 16.02.2025).

10 Kaspersky. *Kaspersky-Cybersicherheitslösungen für Privatanwender und Unternehmen* | Kaspersky. URL: https://kaspersky.com/ (дата звернення: 17.02.2025).

11 The State of AI in Cyber Security - Check Point Research. *Check Point Research.* URL: https://research.checkpoint.com/2025/sate-of-ai-in-cyber-security/ (дата звернення: 18.02.2025).

12 Gartner. *The Gartner Emerging Technologies and Trends in Security & Risk for 2024.* URL: https://www.gartner.com/en/webinar/591158/1321558 (дата звернення: 20.02.2025).

13 Android for enterprise | Android Enterprise | Android Developers. *Android Developers.* URL: https://developer.android.com/work (дата звернення: 01.03.2025).

14 Firebase Cloud Messaging. *Firebase*. URL: https://firebase.google.com/docs/cloud-messaging (дата звернення: 02.03.2025).

15 OWASP Mobile Application Security | OWASP Foundation. *OWASP Foundation, the Open Source Foundation for Application Security | OWASP Foundation.* URL: https://owasp.org/www-project-mobile-app-security/ (дата звернення: 03.03.2025).

16 Android security best practices | Android Open Source Project. Android Open Source Project. URL: https://source.android.com/docs/security/best-practices (дата звернення: 07.03.2025).

17 Flutter documentation. *Docs | Flutter.* URL: https://docs.flutter.dev/ (дата звернення: 14.03.2025).

18 Firebase Documentation. *Firebase*. URL: https://firebase.google.com/docs (дата звернення: 14.03.2025).

19 Privacy changes in Android 10 | Android Developers. *Android Developers.* URL: https://developer.android.com/about/versions/10/privacy/changes (дата звернення: 20.03.2025).

20 General Data Protection Regulation (GDPR) Compliance Guidelines. *GDPR.eu*. URL: https://gdpr.eu/ (дата звернення: 24.03.2025).

ДОДАТКИ

ДОДАТОК А

Технічне завдання

Система для віддаленого видалення конфіденційних даних на Android-пристроях призначена для швидкого захисту інформації у випадках втрати, крадіжки або загрози компрометації. Вона забезпечує оперативне реагування та надійне знищення чутливих даних, запобігаючи їх подальшому несанкціонованому використанню.

1. ПІДСТАВИ ДЛЯ РОЗРОБКИ

Проєкт реалізується в рамках виконання кваліфікаційної роботи та відповідає актуальним потребам у сфері інформаційної безпеки. З огляду на зростаючу кількість витоків конфіденційної інформації через втрату або компрометацію мобільних пристроїв, виникає необхідність у створенні надійної системи, яка дозволить дистанційно видаляти дані з пристрою, незалежно від його фізичного місцезнаходження чи наявності мережевого підключення.

2. ПРИЗНАЧЕННЯ РОЗРОБКИ

Система призначена для захисту персональних і корпоративних даних на мобільних пристроях, які працюють під керуванням операційної системи Android. Продукт може бути використаний як звичайними користувачами, так і в корпоративному середовищі з великою кількістю пристроїв. Основна функція — забезпечення дистанційного, гарантовано безповоротного видалення даних у разі втрати, крадіжки або загрози несанкціонованого доступу до пристрою.

3. ВИМОГИ ДО ПРОГРАМИ ЧИ ПРОГРАМНОГО ПРОДУКТУ

3.1 Функціонал системи

- реєстрація та авторизація користувача;

- прив’язка пристрою до облікового запису;

- отримання та обробка push-сповіщень через Firebase Cloud Messaging;

- вибіркове видалення даних (медіафайли, контакти, повідомлення, кеш додатків, облікові записи);

- повне видалення даних (factory reset);

- ведення логів операцій видалення та аудиту;

- підтвердження доставки та виконання команд.

3.2 Вимоги до надійності

- надійність системи повинна складати не менше 95% при нестабільному мережевому з’єднанні;

- підтримка відкладеного виконання команд у разі офлайн-стану пристрою;

- час обробки команди в онлайн-режимі — до 30 секунд;

- захищеність від повторного виконання однієї і тієї ж команди.

3.3 Умови експлуатації

- сумісність із Android-пристроями версії 6.0 і вище;

- робота у фоновому режимі з мінімальним використанням ресурсів;

- можливість роботи при частковому обмеженні мережевого трафіку або батареї;

- підтримка широкого спектру моделей і брендів смартфонів.

3.4 Вимоги до програмної документації

Програмна документація повинна містити:

- технічне завдання;

- інструкцію користувача;

- кваліфікаційну роботу з поясненням архітектури, логіки функціонування, алгоритмів і тестування.

4. ТЕХНІКО-ЕКОНОМІЧНІ ПОКАЗНИКИ

Система реалізована з використанням безкоштовних фреймворків і сервісів (Flutter, Firebase Free Tier), що дозволяє значно зменшити витрати на розробку. Інтеграція не потребує придбання платних ліцензій або спеціального обладнання. Економічна доцільність впровадження системи виправдана мінімізацією ризиків втрати чутливої інформації та витрат на її відновлення.

5. СТАДІЇ І ЕТАПИ РОЗРОБКИ

1. **Аналітичний етап** — вивчення загроз, визначення вимог, аналіз аналогів.
2. **Проєктування архітектури** — розробка клієнт-серверної структури та логіки обміну даними.
3. **Розробка програмного забезпечення** — реалізація інтерфейсу, обробки команд, видалення даних, логіки безпеки.
4. **Тестування** — перевірка роботи системи на різних пристроях, включаючи офлайн-сценарії.
5. **Документування** — підготовка технічної, пояснювальної та інструкційної документації.

6. ПОРЯДОК КОНТРОЛЮ І ПРИЙМАННЯ

Приймання результатів надання послуг здійснюється в процесі захисту кваліфікаційної роботи.

ДОДАТОК Б

Інструкція користувачу

Цей користувацький посібник призначено для ознайомлення з використанням мобільної системи віддаленого видалення конфіденційних даних на Android-пристроях. У ньому наведено основні етапи реєстрації, взаємодії з контактами та процесу надсилання команд на видалення даних.

1. Реєстрація та вхід у систему

При запуску додатку з’являється екран входу з можливістю створення нового облікового запису. Натисніть кнопку "Зареєструватися". Виконайте процес реєстрації (рис. Б.1):

- заповніть форму реєстрації, вказавши ім’я, email та пароль;
- пароль повинен містити не менше 6 символів;
- після натискання кнопки "Зареєструватися", обліковий запис буде створено.

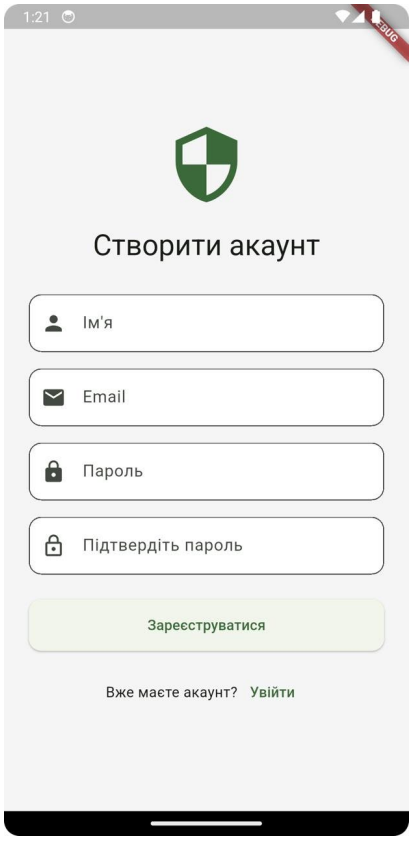

Рисунок Б.1 - Форма реєстрації

Користувачі, які вже мають обліковий запис, можуть авторизуватися через ту ж форму входу (рис. Б.2):

- введіть email та пароль, які використовувалися при реєстрації;
- натисніть кнопку "Увійти";
- після входу відбувається перенаправлення на головну сторінку програми.

Рисунок Б.2 - Форма логінації

2. Робота з контактами

На головному екрані доступні два списки: додані контакти та користувачі, які додали вас (рис. Б.3).

- для додавання нового контакту натисніть кнопку "+";
- введіть email користувача, якого хочете додати;
- контакт з’явиться зі статусом "очікує підтвердження".

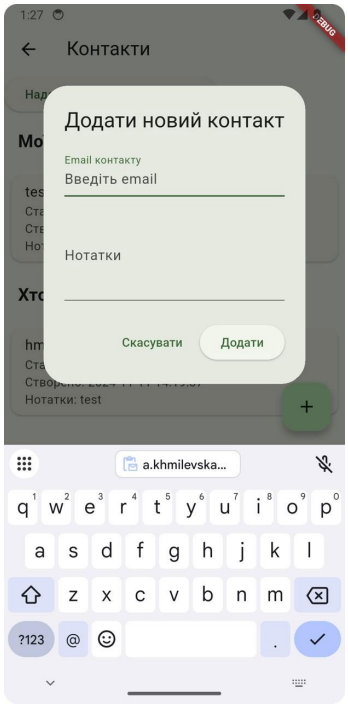

Рисунок Б.3 - Форма додавання нового контакту

Коли інший користувач додає вас у контакти (рис. Б.4):

- ви побачите відповідний запис у розділі "Хто додав мене";
- підтвердіть або відхиліть запит у меню контакту;
- після взаємного підтвердження стає можливою відправка команд на видалення.

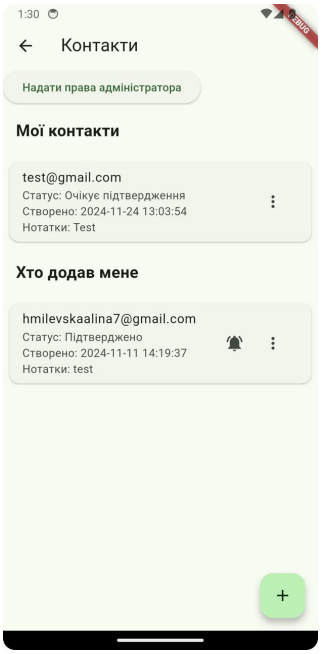

Рисунок Б.4 - Головна сторінка

3. Відправка команди на видалення даних (рис. Б.5):

- для підтверджених контактів доступна кнопка у вигляді дзвіночка;
- натисніть на неї, щоб відправити push-сповіщення на пристрій користувача;
- система повідомить про успішну відправку та виконання команди.

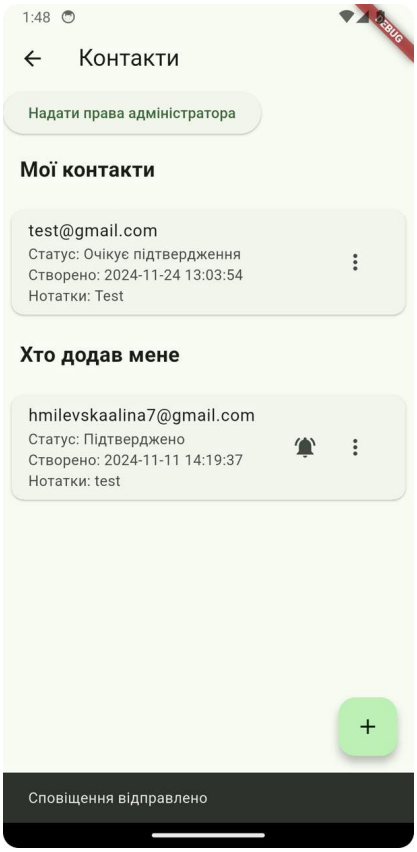

Рисунок Б.5 - Повідомлення про успішну відправку команди

4. Видалення даних на пристрої

Коли команда надходить на пристрій відображається push-сповіщення про ініціацію процесу видалення (рис. Б.6):

- всі медіафайли (наприклад, з галереї) видаляються;
- після завершення процесу галерея відображає порожній список (рис. Б.7).

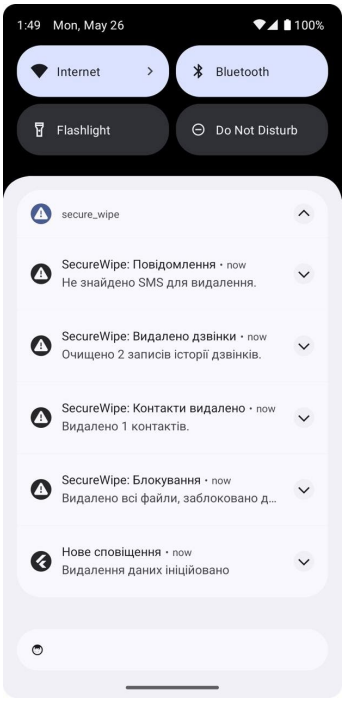

Рисунок Б.6 - Сповіщення під час видалення даних

Рисунок Б.7 - Галерея після видалення даних

Цей посібник дозволить вам ефективно користуватися функціями системи та забезпечити захист конфіденційної інформації в критичних ситуаціях.

АНОТАЦІЯ

Хмілевська А.В. Розробка комплексної системи захисту інформації на мобільних пристроях з функцією віддаленого видалення даних.

Кваліфікаційна робота на здобуття освітнього ступеня «бакалавр» за спеціальністю 122 Комп'ютерні науки. Волинський національний університет імені Лесі Українки, Луцьк, 2025 р.

У роботі розглянуто питання інформаційної безпеки мобільних пристроїв, зокрема ризики, пов'язані з витоком даних у разі їх втрати або викрадення. Запропоновано рішення у вигляді мобільного застосунку, який дозволяє здійснювати віддалене видалення чутливої інформації з пристрою користувача. Система реалізована на основі клієнт-серверної архітектури з використанням сучасних інструментів розробки мобільних застосунків і хмарних сервісів.

Проведено аналіз сучасних загроз безпеці даних на мобільних пристроях, визначено технічні та функціональні вимоги до системи, розроблено архітектуру рішення та реалізовано ключові функції, серед яких: віддалене видалення медіафайлів, контактів, журналу викликів, а також збережених файлів. Особливу увагу приділено безпечній передачі команд та дотриманню вимог сучасних стандартів захисту інформації.

Під час тестування було перевірено працездатність системи на різних версіях Android. Робота має прикладний характер і може бути використана для створення корпоративних політик мобільної безпеки або особистого захисту даних.

Ключові слова: мобільна безпека, віддалене видалення, Flutter, Firebase, інформаційний захист, Android.