НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ
І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

УДК 004.9-047.36

«ПОГОДЖЕНО»	**«ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ»**
Декан факультету інформаційних технологій	Завідувач кафедри комп'ютерних наук
Глазунова О.Г., д.п.н., професор	Голуб Б.Л., к.т.н., доцент
_____	_____
____ _____ 2021 р.	____ _____ 2021 р.

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему «Система аналізу загроз інформаційним ресурсам мобільних пристроїв»

Спеціальність 122 «Комп'ютерні науки»

Освітня програма «Інформаційні управляючі системи та технології»

Орієнтація освітньої програми освітньо-професійна

Гарант освітньої програми «Інформаційні управляючі системи та технології»

д.т.н, доцент _____ Бондаренко В.Є.

Керівник магістерської кваліфікаційної роботи

к.т.н., доцент _____ Пархоменко І.І.

Виконав _____ Якимов О. С.

КИЇВ-2021

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних наук
Голуб Б.Л., к.т.н., доцент
"29" жовтня 2020 року

З А В Д А Н Н Я

ДО ВИКОНАННЯ МАГІСТЕРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ СТУДЕНТУ

Якимову Олексію Сергійовичу
Спеціальність 122 «Комп'ютерні науки»
Освітня програма Інформаційні управляючі системи та технології
Орієнтація освітньої програми освітньо-професійна
Тема магістерської кваліфікаційної роботи «Система аналізу загроз інформаційним ресурсам мобільних пристроїв»
затверджена наказом ректора НУБіП України від "29" жовтня 2020р. №1634 «С»
Термін подання завершеної роботи на кафедру "10" грудня 2021р.
Вихідні дані до магістерської кваліфікаційної роботи
1) Дані про найбільш уразливі програми, які використовують злочинці під час кібератак
2) Дані про ключові слова-мітки, наявність яких збільшує ступінь ймовірної загрози
Перелік питань, що підлягають дослідженню:

№ з/п	Питання, що підлягає дослідженню	Строк виконання	Примітка
1.	Аналіз предметної області.	29.10.2020 – 30.09.2021	
2.	Моделювання системи	30.09.2021 – 15.10.2021	
3.	Розробка системи	15.10.2021 – 01.11.2021	
4.	Результати дослідження	01.11.2021 – 30.11.2021	
5.	Попередній захист	30.11.2021	
6.	Захист	14.12.2021	

Дата видачі завдання "29" жовтня 2020 року

Керівник магістерської кваліфікаційної роботи _____ Пархоменко І. І.
(підпис)

Завдання прийняв до виконання _____ Якимов О.С.
(підпис)

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

БД – база даних

ГБ – гігабайт

ГГц – гігагерци

ОЗУ - оперативно запам'ятовуючий пристрій

ОС – операційна система

СУБД– система управління базою даних

API (ApplicationProgramming Interface) - програмний інтерфейс додатку

IDE (Integrated Development Environment) - інтегроване середовище розробки

IOS (Iphone Operation System) – операційна система від компанії Apple

RAW - формат цифрових файлів зображення

SDK (Software Development Kit) - комплект засобів розробки

SHA (Secure Hash Algorithm) - алгоритм криптографічного хешування.

SSL (Secure Sockets Layer) - рівень захищених сокетів

SQL (Structured query language) – мова структурованих запитів

UML (Unified Modeling Language) – уніфікована мова моделювання

XML (Extensible Markup Language) – розширювана мова розмітки

ВСТУП

На сьогоднішній день тема захисту даних на мобільних пристроях актуальна як ніколи, оскільки, смартфон став мультимедійним центром і, своєрідною, кишеньковою робочою станцією. Кіберзлочинці з кожним днем стають все кмітливішими та небезпечнішими, знаходять обхідні шляхи для заходів безпеки та проникають у безпечні мережі. На жаль, люди повинні бути так само занепокосні: навіть якщо ви переглядаєте Інтернет у захищеному браузері та використовуєте програмне забезпечення для виявлення загроз, підступна програма може виявитися надто легкою, щоб підступна програма з'явилася на вашому девайсі.

На мобільних пристроях, які ми використовуємо повсякденно, зазвичай міститься важлива інформацію. Історія перегляду сторінок в браузері, історія дзвінків, фотографії, календарі, текстові і голосові повідомлення, адресні книги та інші корисні речі – це все може створити ланцюжок неприємностей на втраченому або викраденому смартфоні. Необхідно знати, в якому місці на мобільному телефоні зберігається важлива інформація та дані онлайн сервісів, до яких існує автоматичний доступ. Доступ до цих даних може спричинити проблеми не тільки щодо власника пристрою, але і кожного, хто опиниться в у вхідних повідомленнях, адресній книзі або у фотоальбомі.

На сьогоднішній день, не так багато користувачів смартфонів приховують або ускладнюють доступ до певної інформації, тому, у кого є фізичний доступ до пристрою, може бути нескладно отримати її.

Щороку кількість шкідливих додатків для мобільних пристроїв подвоюється, тому, практично, кожен користувач телефону або планшету мав справу з негативними наслідками діяльності шкідливого ПЗ. Найчастіше, звичайно, це нав'язлива реклама, яка може з'являється в будь який момент,

вискакувати при розблокуванні пристрою або навіть замінити фон робочого столу. Більш того, віруси можуть підміняти рекламу, яка з'являється в різних безкоштовних додатках, а також перенаправляти браузер користувача на різні шкідливі сайти.

Безсумнівно, нав'язлива реклама та віруси – становлять небезпеку для вашого мобільного девайсу, планшета або іншого пристрою. Тому необхідно швидше провести моніторинг, виявити і видалити вірус.

Метою дослідження є пошук вірогідних загроз інформаційним ресурсам мобільних пристроїв, забезпечення конфіденційності даних на мобільному пристрої.

Об'єктом дослідження є процеси виявлення загроз інформаційним ресурсам мобільних пристроїв

Предметом дослідження є система моніторингу загроз інформаційним ресурсам мобільних пристроїв.

1 СИСТЕМНИЙ АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Постановка задачі

Основною задачею є розробка система аналізу загроз інформаційним ресурсам мобільних пристроїв. Кожен користувач смартфону повинен приймати необхідні заходи задля підвищення рівня безпеки. Зловмисникам, звичайно, простіше зламати пристрій, який не захищений від несанкціонованого доступу. Не завжди, навіть, допомагає встановлення складного паролю або графічного ключа при розблокуванні телефону, проникнення загрози може відбутися при завантаженні сумнівного додатку або файлу з месенджера.

Розроблена систему аналізу загроз інформаційним ресурсам мобільних пристроїв була створена для покращення рівня безпеки, підвищення конфіденційності даних на мобільному пристрої.

Основні переваги такої інформаційної системи:

- сканування пристрою на наявність шкідливого ПЗ;
- переміщення підозрюваних файлів до карантину;
- можливість додавання файлів до вийнятків;
- перегляд файлів, що знаходяться на карантині та файлів-виключень.

1.2 Огляд інформаційних джерел та існуючих рішень

Операційні системи Apple мають заслужену репутацію за безпеку та протидію злому. Вони не ідеальні, але суворий контроль Apple означає, що їх можна жорстко заблокувати. Windows і Android, з іншого боку, не були побудовані на принципі безпеки, який приніс нам iOS. У вас, безсумнівно, є принаймні антивірус, який захищає ваші Windows, але як щодо ваших телефонів і планшетів Android? Вони настільки ж чутливі і настільки ж вразливі. Вам потрібен антивірус Android або, краще, повномасштабний пакет безпеки, який

включає антивірус, захист від крадіжки тощо. Захистити гаджети Android потрібно з першого ж дня користування пристроєм.

Захист Android не існує на порожньому місці. Усі продукти, перелічені тут, є міжплатформними рішеннями, із захистом, доступним принаймні для Windows, macOS та Android. Більшість із них також пропонують певну форму захисту iOS, хоча з набором функцій, обмеженим закритістю операційної системи Apple.

Зауважте, що рейтинги стосуються продукту в цілому, на всіх платформах. Вони не обов'язково відображають якість продукту Android. Погані результати тесту на Windows можуть призвести до зниження загальної оцінки продукту, навіть якщо його видання Android пройшло нормально.

1.2.1 Захист від шкідливого програмного забезпечення.

Усі продукти Android включають антивірусний компонент, який сканує нові програми, а також пропонує сканування на вимогу. Усі вони також пропонують певну форму безпечного перегляду, щоб уберегти вас від перегляду URL-адрес, які можуть спробувати запровадити зловмисне програмне забезпечення, або шахрайських сайтів, які можуть змусити вас надати ваше ім'я користувача та пароль для сторінки входу, яку вони імітують.

Цей захист на основі користувачів дозволяє Android створити «пісочницю програми». Кожній програмі для Android призначається унікальний ідентифікатор користувача, і кожен виконується як окремий процес. Таким чином, кожна програма забезпечується на рівні процесу через ядро Linux, яке не дозволяє програмам взаємодіяти один з одним і надає їм лише обмежений доступ до операційної системи Android. Це дає користувачеві контроль доступу на основі дозволів, і він/вона отримує список дій, які буде виконувати програма Android, і те, що їй потрібно для їх виконання, ще до того, як додаток навіть завантажиться. Те ж саме стосується дозволів файлової системи – кожна програма (або користувач) має власні файли, і якщо розробник явно не надає файли іншій програмі Android, файли, створені однією програмою, не можуть бути прочитані або змінені іншою.

Усі додаткові технічні функції безпеки Android розроблені так, щоб бути просто представленими користувачеві, що означає, що ними можна легко керувати через інтерфейс. Прості методи покращення безпеки вашого пристрою Android можуть включати: використання пароля або PIN-коду, блокування телефону після певного періоду бездіяльності, увімкнення лише бездротових з'єднань, які ви використовуєте, а також встановлення лише програм Android, яким ви довіряєте та які особисто перевірили.

Google також допускає на свій ринок лише перевірені та перевірені безпечні програми Android, що означає, що користувач має менше шансів встановити шкідливий додаток. Крім того, система безпеки Android пропонує користувачеві дозволити встановлення програми, що означає, що неможливо віддалено встановити та запустити програму. Крім того, користувачі можуть забезпечити безпеку свого пристрою Android, регулярно встановлюючи оновлення системи.

Усі ці програми також перевіряють встановлені програми на наявність потенційних проблем із конфіденційністю. Як правило, вони позначають програми, які мають дозвіл робити такі дії, як перегляд ваших контактів, сканування журналів викликів, визначення вашого місцезнаходження або надсилання текстів. Якщо комунікаційній програмі потрібен доступ до контактів, це має сенс. Однак якщо дурна гра хоче перевіряти вашу особисту інформацію, подумайте про її видалення.

Усі, крім одного, включають захист від крадіжки втраченого або вкраденого пристрою. Найважливішим для захисту від крадіжки є Norton, який відмовився від цієї можливості у 2019 році. Ви можете знайти місцезнаходження свого пристрою на карті. Якщо ви щойно розклали його по дому, ви можете ввімкнути гучний сигнал, щоб допомогти вам його знайти. Ви можете заблокувати телефон, щоб не допустити злодія до ваших програм і даних. І якщо ви вирішите, що ніколи не отримаєте пристрій назад, ви можете віддалено стерти його. Усі програми дозволяють керувати функціями захисту від крадіжки за допомогою онлайн-консолі. Більшість із них пропонують можливість ініціювати

заходи проти крадіжки за допомогою закодованих текстових повідомлень, і багато з них також приховано фотографують того, хто використовує ваш пристрій.

Антивірус і захист від крадіжки є основними компонентами будь-якої програми безпеки Android, але деякі виходять за рамки основ. Загальні бонусні функції включають резервне копіювання ваших контактів і фотографій, монітор батареї, щоб показувати, які програми вбивають заряд батареї, і засіб для вбивства завдань, щоб відправити ці заряди акумулятора. Деякі програми попереджають про підключення до незахищеної мережі Wi-Fi. Bitdefender, Kaspersky і McAfee дозволяють підключити телефон до Android Wear, тому, якщо ви відійдете від телефону, годинник може нагадати вам, що потрібно його взяти.

Як зазначалося, майже всі ці інструменти безпеки сканують встановлені додатки та повідомляють про ті, які можуть бути ризиками для конфіденційності. Norton і Trend Micro виводять цю навичку на новий рівень, повідомляючи про програми, коли ви переглядаєте їх у Play Store, тож ви можете уникнути завантаження тих, які можуть бути проблематичними. `1]

Окремі додатки для проведення сканування системи мобільного пристрою встановлюють (дуже часто пропонують) встановлення додаткового рекламного або подібного ПЗ, а також, зазвичай, використовують рекламу у своєму застосунку.

1.3 Типи шкідливого програмного забезпечення

Користувачі персональних комп'ютерів часом вважають шкідливі програми серйозною загрозою, хоча часто стають жертвами крадіжки облікових даних чи блокування комп'ютера з вимогою викупу. Бізнес, навпаки, говорить про інфекції як суттєву загрозу своїй діяльності. Поширення отримують шкідливі програми, які потрапляють до пристроїв інтернету речей. Так, компанія із Британії створила вірус-вимагач для термостата, підключеного до Wi-Fi.

Отримавши контроль над обладнанням, він здатний опустити температуру до критичної позначки та вимагати грошей. Докладніше про нову шкідливу програму можна прочитати у статті «Віруси-здирники дісталися термостатів». Немає абсолютного захисту від інфекцій, але зменшити ризик реалізації загрози можна. Для цього необхідно встановлювати нові версії операційних систем, стежити за оновленням усіх програм, використовувати антивірусні рішення від надійних виробників, не допускати до ПК сторонніх осіб, не відкривати підозрілі посилання, листи та файли, виконувати інші запобіжні заходи.

Хробаки. Хробаки поширюються через вразливості програмного забезпечення або фішингові атаки. Як тільки хробак встановлюється в пам'ять вашого комп'ютера, він починає заражати всю машину, а в деяких випадках… всю вашу мережу.

Залежно від типу хробака та ваших заходів безпеки вони можуть завдати серйозної шкоди. Ці паразити можуть:

- Змінювати та видаляти файли
- Вводити шкідливе програмне забезпечення на пристрої
- Повторювати себе знову і знову, щоб виснажити системні ресурси
- Вкрасти ваші дані
- Встановити зручний бекдор для хакерів

Вони можуть швидко заразити велику кількість комп'ютерів, споживаючи пропускну здатність і перевантажуючи ваш веб-сервер під час роботи.

Віруси. На відміну від хробаків, для роботи вірусам потрібна вже інфікована активна операційна система або програма. Віруси зазвичай прикріплюються до виконуваного файлу або текстового документа. Більшість людей, ймовірно, знають, що розширення файлу .exe може призвести до проблем, якщо воно не з надійного джерела. Але існують сотні інших розширень файлів, які позначають виконуваний файл.

Зазвичай вірус, що поширюється через заражені веб-сайти, обмін файлами або завантаження вкладень електронної пошти, буде бездіяльним, доки

не буде активовано заражений файл хоста або програма. Як тільки це станеться, вірус здатний розмножуватися і поширюватися через ваші системи.

Боти та ботнети. Бот – це комп'ютер, заражений зловмисним програмним забезпеченням, тому хакер може віддалено керувати ним. Цей бот (він же комп'ютер-зомбі) може потім використовуватися для запуску нових атак або стати частиною колекції ботів (він же ботнет).

Ботнети користуються популярністю серед хакерських показів (чим більше ботів ви збираєте, тим сильніший ви хакер) і кіберзлочинців, які поширюють програму-вимагач. Ботнети можуть включати мільйони пристроїв, коли вони поширюються непомітно. Ботнети допомагають хакерам виконувати всілякі шкідливі дії, зокрема:

- DDoS атаки
- Keylogging, скріншоти та доступ до веб-камери
- Поширення інших типів шкідливих програм
- Розсилка спаму та фішингових повідомлень

Троянські коні. Троянський кінь – це шкідлива програма, яка маскується під легітимний файл. Оскільки він виглядає надійним, користувачі завантажують.

Самі трояни є дверним прорізом. На відміну від хробака, для роботи їм потрібен господар. Як тільки ви отримаєте троян на вашому пристрої, хакери можуть використовувати його для:

- Видалення, зміни та захоплення даних
- Захоплення вашого пристрою як частину ботнету
- Шпигування за вашим пристроєм
- Отримання доступу до вашої мережі

Програми-вимагачі. Програма-вимагач забороняє або обмежує доступ до ваших власних файлів. Потім він вимагає оплати (зазвичай криптовалютами) в обмін на те, що дозволить вам повернутися. У травні 2017 року атака програмного забезпечення-вимагача поширилася на 150 країн і лише за один день скомпрометувала понад 200 тисяч комп'ютерів. Атака під влучною назвою

WannaCry завдала збитків, які оцінюються від сотень мільйонів до мільярдів доларів.

WannaCry торкнувся операційних систем MS, на яких не було встановлено останнього виправлення для відомої вразливості. Щоб зменшити ризик атак програм-вимагачів:

- Завжди оновлюйте свою операційну систему
- Підтримуйте своє антивірусне програмне забезпечення в актуальному стані
- Створіть резервні копії найважливіших файлів
- Не відкривайте вкладення з невідомих джерел (WannaCry поширювався через вкладення .js)

Рекламне програмне забезпечення та шахрайство. Рекламне програмне забезпечення є одним із найбільш відомих типів шкідливих програм. Він обслуговує спливаючі вікна та медійні оголошення, які часто не мають для вас жодного відношення. Деякі користувачі миряться з певними типами рекламного ПЗ в обмін на безкоштовне програмне забезпечення (наприклад, ігри). Але не всі рекламні програми однакові. У кращому випадку це дратує і сповільнює роботу вашої машини. У гіршому, реклама посилається на сайти, де шкідливі завантаження чекають нічого не підозрюючих користувачів. Рекламне програмне забезпечення також може доставляти шпигунське програмне забезпечення, яке часто легко зламати, що робить пристрої, на яких воно встановлено, м'якою мішенню для хакерів, фішерів і шахраїв.

Шпигунське програмне забезпечення. Шпигунське програмне забезпечення таємно записує вашу онлайн-активність, збирає ваші дані та збирає особисту інформацію, таку як імена користувачів, паролі та звички серфінгу.

Шпигунське програмне забезпечення – це поширена загроза, яка зазвичай поширюється як безкоштовне або умовно-безкоштовне програмне забезпечення, яке має привабливу функцію на передньому плані з прихованою місією, що виконується у фоновому режимі, яку ви ніколи не помітите. Його часто

використовують для крадіжки особистих даних і шахрайства з кредитними картками.

Потрапивши на ваш комп'ютер, шпигунське програмне забезпечення передає ваші дані рекламодавцям або кіберзлочинцям. Деякі шпигунські програми інсталюють додаткове зловмисне програмне забезпечення, яке вносить зміни у ваші налаштування.

Спам і фішинг. Фішинг – це тип атаки соціальної інженерії, а не тип зловмисного програмного забезпечення. Але це поширений метод кібератаки. Фішинг успішний, оскільки надіслані електронні листи, текстові повідомлення та створені веб-посилання виглядають так, ніби вони з надійних джерел. Їх надсилають злочинці для шахрайського отримання особистої та фінансової інформації.

Деякі з них дуже складні і можуть обдурити навіть найдосвідченіших користувачів. Особливо у випадках, коли обліковий запис електронної пошти відомого контакту зламано, і здається, що ви отримуєте вказівки від свого начальника або IT-колег. Інші менш досконалі й просто надсилають якомога більше електронних листів із повідомленням про «перевірку реквізитів вашого банківського рахунку». [2]

На Рис. 1 зображені найбільш уразливі програми, які використовуються злочинцями під час кібератак.

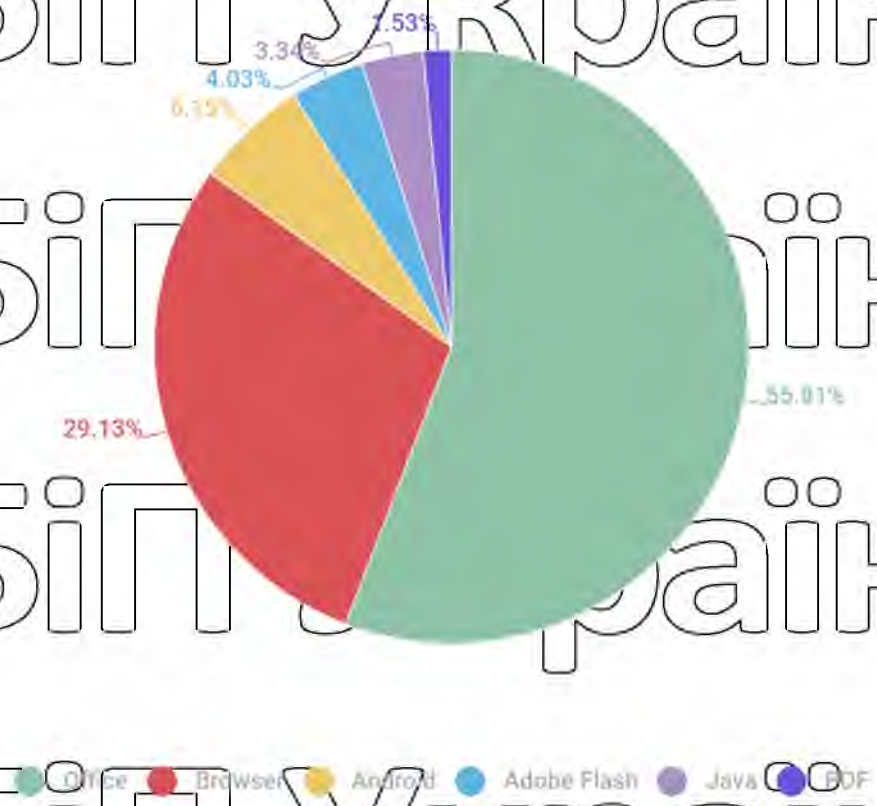

Рисунок 1. Уразливі програми, які використовують злочинці під час кібератак

Другий квартал 2021 року вніс деякі незначні зміни в статистику про експлойти, якими користуються кіберзлочинці. Зокрема, частка експлойтів для Microsoft Office знизилася до 55,81% від загальної кількості загроз такого типу. І навпаки, частка експлойтів, що атакують популярні браузери, зросла приблизно на 3 п.п. до 29,13%.

Об'єкти впливу. Атаки шкідливих програм поширюються практично на всіх користувачів Інтернету. Мета впливу залежить від типу зловмисника: хуліган, дрібний злодій чи кіберзлочинець. Відповідним чином розрізняються і наслідки: одна інфекція просто заважає нормально працювати з комп'ютером, інша — призводить до фінансових збитків, третя — закінчується витіканням відомостей, що становлять комерційну таємницю. В останні роки від шкідливих програм часто страждають різні компанії та організації — насамперед через свою платоспроможність. Типовою атакою є шифрування, наприклад, бухгалтерської

бази даних та подальшу вимогу заплатити за відновлення цієї критично важливої для бізнесу інформації. Атакам експлойтів, троянів та черв'яків піддаються сервери веб-сайтів, звідки зловмисники крадуть інформацію про клієнтів та користувачів, включаючи дані банківських карток, що загрожує втратою фінансів, баз даних, іншої корпоративної інформації. Об'єктами застосування шкідливих програм є і звичайні користувачі Мережі. Інтерес представляють особисті дані, інформація про банківські рахунки, електронна пошта, паролі доступу до соціальних мереж. Досить часто метою інфекцій стають геймери, що мають велику кількість ігрової валюти та рідкісні артефакти.

2 МОДЕЛЮВАННЯ СИСТЕМИ

2.1 Моделювання предметної області

UML, Unified Modeling Language, — це стандартизована мова моделювання, що складається з інтегрованого набору діаграм, розроблених для допомоги розробникам систем і програмного забезпечення для визначення, візуалізації, конструювання та документування артефактів програмних систем, а також для бізнес-моделювання та інші непрограмні системи. UML являє собою набір найкращих інженерних практик, які виявилися успішними в моделюванні великих і складних систем. UML є дуже важливою частиною розробки об'єктно-орієнтованого програмного забезпечення та процесу розробки програмного забезпечення. UML використовує переважно графічні позначення для вираження дизайну програмних проектів. Використання UML допомагає проектним групам спілкуватися, досліджувати потенційні проекти та перевіряти архітектурний дизайн програмного забезпечення. У цій статті ми дамо вам докладні уявлення про те, що таке UML, історію UML та опис кожного типу діаграми UML, а також приклади UML. [3]

Розроблені діаграми прецедентів, архітектури системи та логічна модель БД відображають процеси, які відбуваються у системі моніторингу загроз інформаційним ресурсам мобільних пристроїв.

2.1.1 Діаграма прецедентів описує функціональні вимоги системи з точки зору варіантів використання. Це модель передбачуваної функціональності системи (варіанти використання) та її середовища (акторів). Варіанти використання дозволяють зв’язати те, що вам потрібно від системи, і те, як система забезпечує ці потреби.

Оскільки це дуже потужний інструмент планування, модель варіантів використання зазвичай використовується на всіх фазах циклу розробки всіма членами команди.

Існує два основних види відносин між компонентами на діаграмах прецедентів.

Відношення розширення (extend relationship) визначає взаємозв'язок прецеденту з прецедентом, можливості якого він може використовувати. Графічно позначається пунктирною стрілкою з позначкою «extend» від доповнює прецеденту до розширюватися;

Відношення включення (include relationship) вказує на включення прецеденту в інший прецедент в якості його складової частини. Один і той же прецедент може бути включений в кілька більших прецедентів. Графічно дане відношення позначається суцільною лінією зі стрілкою, спрямованої від базового прецеденту до такого, що включається з позначкою «include».

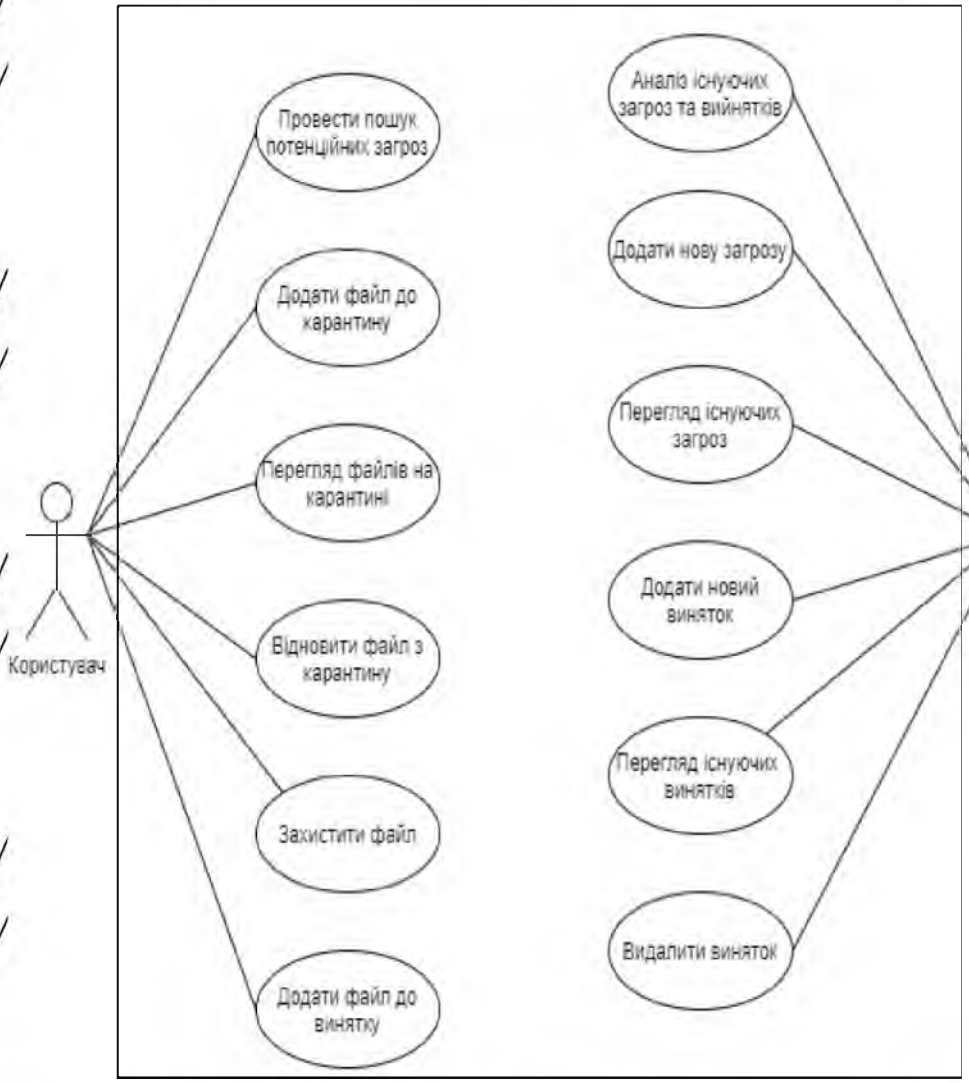

Рисунок 2. Діаграма прецедентів

На Рис. 2 можна побачити відношення між усіма акторами та прецедентами в системі. На діаграмі виділені 2 актори: користувач та адміністратор. Прецеденти дозволяють зрозуміти поведінку розробленої системи та отримати вілповідь на запитання, що має робити система.

2.2 Логічна модель БД

Логічна модель даних — це тип моделі даних, яка детально описує елементи даних і використовується для розвитку візуального розуміння об'єктів даних, атрибутів, ключів і зв'язків. Цей тип моделі унікально не залежить від

конкретної бази даних, щоб створити базову структуру для компонентів семантичного рівня в системах керування даними.

Логічна модель даних описує дані якомога детальніше, незалежно від того, як вони будуть фізично реалізовані в базі даних.

До особливостей логічної моделі даних належать:

- Включає всі сутності та зв'язки між ними.
- Усі атрибути для кожної сутності вказуються.
- Визначаються первинний ключ для кожної сутності.
- Визначаються зовнішні ключі (ключі, що позначають зв'язок між різними сутностями).
- На цьому рівні відбувається нормалізація.

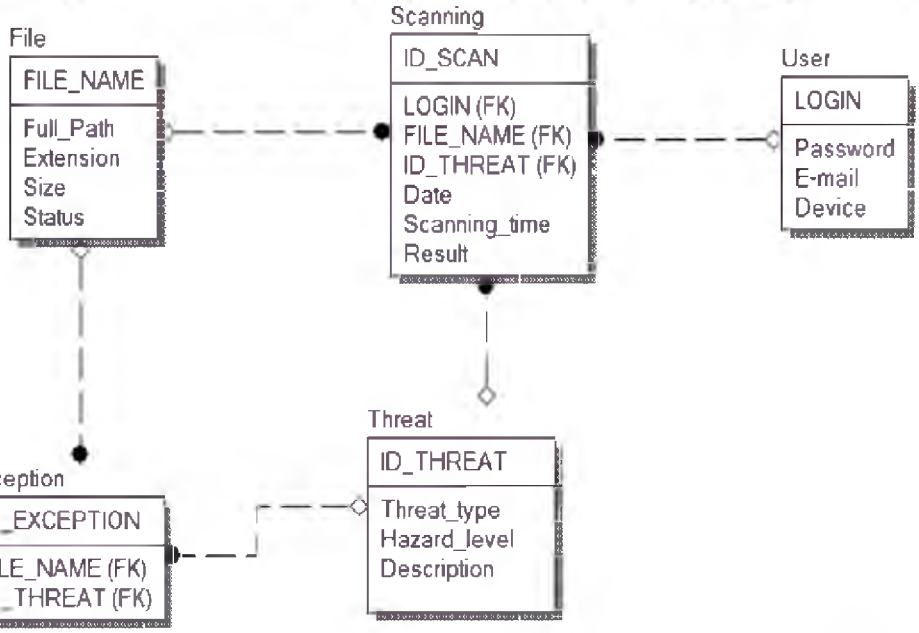

Рисунок 3. Логічна модель БД

На Рис. 3 зображено модель даних, що створена за допомогою програмного продукту ERWin Data Modeler. Всього виділено 5 сутностей: Файл, Сканування, Користувач, Загроза та Виключення.

2.2 Структура сховища даних

OLAP (Online Analytical Processing - аналітична обробка даних в реальному часі) являє собою потужну технологію обробки і дослідження даних. Системи, побудовані на основі технології OLAP, надають практично безмежні можливості по складанню звітів, виконання складних аналітичних розрахунків, побудови прогнозів і сценаріїв, розробці безлічі варіантів планів.

Своє застосування OLAP системи знайшли в багатьох питаннях стратегічного управління організацією: стратегічне планування, імітаційне моделювання зовнішнього і внутрішнього середовища організації, бюджетування, підготовка фінансової звітності, управління ефективністю бізнесу, аналіз роботи, прогнозування розвитку, зберігання даних і звітності.

У загальному вигляді, структура OLAP системи складається з наступних елементів:

- база даних. База даних є джерелом інформації для роботи OLAP системи. Вид бази даних залежить від виду OLAP системи і алгоритмів роботи OLAP сервера. Як правило, використовуються реляційні бази даних, багатовимірні бази даних, сховища даних і т.п.
- користувацькі програми. Цей елемент структури OLAP системи здійснює управління запитами користувачів і формує результати звернення до бази даних (звіти, графіки, таблиці та ін.)
- OLAP сервер. Він забезпечує управління багатовимірної структурою даних і взаємозв'язок між базою даних і користувачами OLAP системи

Застосування OLAP системи дає організації можливість по прогнозуванню та аналізу різних ситуацій, пов'язаних з поточною діяльністю і перспективами розвитку. Ці системи можна розглядати як додаток до систем автоматизації рівня підприємства.

Основними перевагами OLAP системи є:

- узгодженість вихідної інформації і результатів аналізу. При наявності OLAP системи завжди є можливість простежити джерело інформації і визначити логічний зв'язок між отриманими результатами і вихідними даними. Знижується суб'єктивність результатів аналізу.
- управління деталізацією. Деталь представлення результатів може змінюватися в залежності від потреби користувачів. При цьому немає необхідності здійснювати складні настройки системи і повторювати обчислення. Звіт може містити саме ту інформацію, яка необхідна для прийняття рішень.
- проведення багатоваріантного аналізу. Застосування OLAP системи дозволяє отримати безліч сценаріїв розвитку подій на основі набору вихідних даних. За рахунок інструментів аналізу можна змоделювати ситуації за принципом «що буде, якщо».
- виявлення прихованих залежностей. За рахунок побудови багатовимірних зв'язків з'являється можливість виявити і визначити приховані залежності в різних процесах або ситуаціях, які впливають на виробничу діяльність.
- створення єдиної платформи. За рахунок застосування OLAP системи з'являється можливість створити єдину платформу для всіх процесів прогнозування і аналізу на підприємстві. Зокрема, дані OLAP системи, є основою для побудови прогнозів бюджету, прогнозу продажів, прогнозу закупівель, плану стратегічного розвитку та ін.[4]

Сховище даних - це різновид системи управління даними, яка забезпечує підтримку бізнес-аналітики. Сховища даних призначені тільки для виконання запитів і аналізу і зазвичай містять великі обсяги історичних даних. Дані зазвичай надходять в сховище з найрізноманітніших джерел, таких як журнали додатків і додатки транзакцій.

Сховище даних служить для централізації і консолідації великих обсягів даних з різних джерел. Аналітичні інструменти дають можливість організаціям отримувати від власних даних цінні для бізнесу відомості і підвищувати ефективність прийнятих рішень. Згодом в сховище накопичуються записи за минулі періоди, які становлять велику цінність для фахівців з вивчення даних і бізнес-аналітиків. Ці можливості роблять сховища даних "єдиним джерелом перевірених даних компанії."

Організації, які використовують сховище даних для підтримки аналітики та бізнес-аналітики, бачать ряд істотних переваг:

Кращі дані — додавання джерел даних до сховища даних дозволяє організаціям гарантувати, що вони збирають узгоджені та відповідні дані з цього джерела. Їм не потрібно замислюватися, чи будуть дані доступними чи суперечливими, коли вони надходять до системи. Це забезпечує вищу якість даних і цілісність даних для прийняття обґрунтованих рішень.

Швидше прийняття рішень — дані в сховищі мають такі узгоджені формати, що вони готові до аналізу. Він також забезпечує аналітичну силу та більш повний набір даних для прийняття рішень на основі фактичних фактів. Тому особам, які приймають рішення, більше не потрібно відповідати на припущення, неповні дані або дані низької якості, і вони ризикують отримувати повільні та неточні результати.

На технічному рівні сховище даних періодично отримує дані з цих програм і систем; потім дані проходять процес форматування та імпорту, щоб відповідати даним, які вже є на складі. Сховище даних зберігає ці оброблені дані, тому вони готові для доступу особам, які приймають рішення. Частота витягування даних, форматування даних тощо залежить від потреб організації.[5]

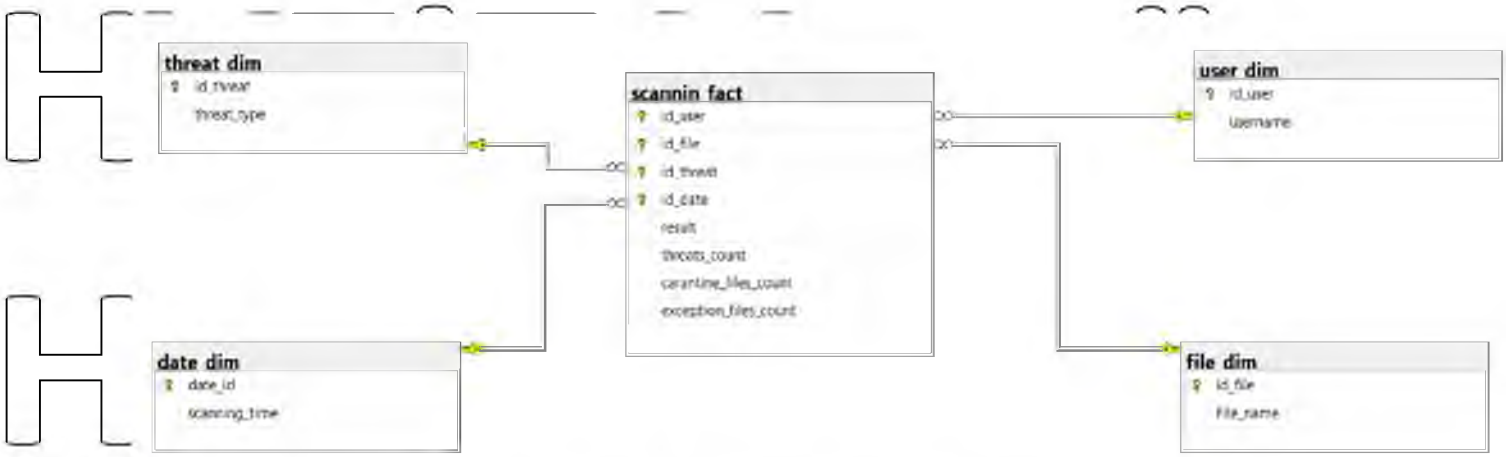

Рисунок 4. Схема сховища даних

На Рис. 4 зображено сховище даних типу «зірка». Воно складається з таблиці фактів Сканування та 4 таблиць вимірів: Користувач, Файл, Загроза, Дата.

4.1 Діаграма розгортання

В UML діаграми розгортання моделюють фізичну архітектуру системи. Архітектура системи відображає те, як взаємозв'язані програмна сторона з апаратними компонентами в системі та фізичний розподіл обробки даних.

Діаграми розгортання, які ви зазвичай готуєте на етапі розробки впровадження, показують фізичне розташування вузлів у розподіленій системі, артефакти, які зберігаються на кожному вузлі, а також компоненти й інші елементи, які реалізують артефакти. Вузлами виступають апаратні пристрої, такі як комп'ютери, принтери або мобільні девайси, а також інші пристрої, які підтримують середовище виконання системи. Шляхи зв'язку та відносини розгортання моделюють з'єднання в системі.[6]

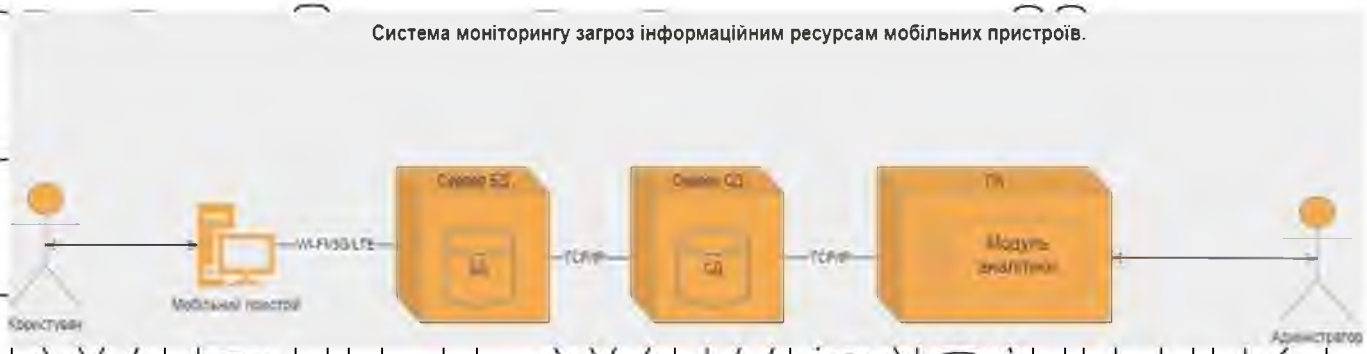

Рисунок 5. Топологія системи

Загальну архітектуру системи можна побачити на Рис. 5. На ній виділено основного актора (користувач, який встановлює додаток на мобільний пристрій та працює з ним, виконуючи сканування пристрою використовуючи БД вірусів) та Адміністратора (контролює оновлення БД вірусів, аналізує кількісні показники додатку та приймає рішення щодо їх подальшого зменшення або збільшення).

3 РОЗРОБКА СИСТЕМИ

3.1 Вибір методів та засобів для реалізації інформаційного забезпечення системи

Для створення сховища даних проекту «Система моніторингу загроз інформаційним ресурсам мобільних пристроїв» обрано систему управління реляційною базою даних – Microsoft SQL Server.

Однією з основних цілей Microsoft SQL Server є забезпечення високого рівня безпеки вашої бази даних, особливо за допомогою служби адміністрування бази даних Microsoft SQL Server. Це програмне забезпечення дозволяє працювати зі структурою таблиць, яка з'єднує функції та елементи даних, що допомагає захистити дані, які у вас є. Безпека та цілісність - ключові параметри для баз даних, які містять інформацію про клієнта та інші конфіденційні дані.

На відміну від іншого програмного забезпечення для керування базами даних, установка та налаштування Microsoft SQL Server є легшими. Для встановлення програмного забезпечення не потрібно мати спеціальний набір інструментів, а оновлення відбуваються повністю автоматично. Ви також можете встановити інші компоненти, щоб змінити програмне забезпечення для вашого бізнесу без складних процесів. Отже, якщо ви шукаєте програмне забезпечення для керування базами даних, яке забезпечує зручність, Microsoft SQL Server — це шлях.

З Microsoft SQL Server вам не потрібно мати інше сховище даних з тієї ж бази даних, якщо ви використовуєте інший пристрій. Це дозволяє легко й ефективно керувати даними з мінімальними проблемами та обслуговуванням. Таким чином, ви можете заощадити час і працювати над іншими важливими аспектами вашого бізнесу.

У разі відключення живлення або вимкнення сервера дані можуть пошкодитися, що створює велику проблему для компаній, які майже не

зберігають резервних коній. Microsoft SQL Server усуває ризик втрати даних завдяки функціям відновлення та відновлення даних. Як результат, ви матимете більший спокій, знаючи, що ваші дані захищені за допомогою кешування, файлів журналів і частого резервного копіювання, незалежно від того, що може статися з вашим сервером.

Для взаємодії додатку з базою даних використовуються скриптова мова PHP. На відмінну від інших скриптових мов, PHP-скрипти виконуються на сервері та генерують HTML, який пересилається клієнту. Стандартні бібліотеки PHP мають усі необхідні функції для взаємодії з MS SQL.

3.1.1 Опис BI та створення проекту служби SSAS

Business Intelligence (BI) поєднує в собі бізнес-аналітику, аналіз даних, візуалізацію даних, інструменти та інфраструктуру даних, а також найкращі методи, щоб допомогти організаціям приймати рішення на основі даних. На практиці нам знайомо, що мати сучасну бізнес-аналітику, означає мати вичерпне уявлення про дані своєї організації та використовуєте ці дані для стимулювання змін, усунення неефективності та швидкої адаптації до змін ринку чи пропозиції.

Важливо зазначити, що це дуже сучасне визначення BI, і BI мав задушену історію як модне слово. Традиційна Business Intelligence, великі літери і все таке, спочатку виникла в 1960-х роках як система обміну інформацією між організаціями. У 1980-х роках він отримав подальший розвиток разом із комп'ютерними моделями для прийняття рішень і перетворення даних у інтуїцію, перш ніж стати специфічною пропозицією команд BI з рішеннями для обслуговування, що залежать від IT. Сучасні рішення BI віддають перевагу гнучкому аналізу самообслуговування, керованим даним на надійних платформах, уповноваженим бізнес-користувачам і швидкості отримання інформації. [7].

Для реалізації програмної сторони системи моніторингу загроз інформаційним ресурсам мобільних пристроїв було використано IDE Android Stuidio.

Android Studio – офіційне середовище інтегрованого розвитку (IDE) для розробки додатків для Android, засноване на IntelliJ IDEA. Крім потужного редактора коду та інструментів для розробників IntelliJ, Android Studio пропонує ще більше функцій, що підвищують вашу продуктивність при створенні програм для Android, таких як:

- гнучка система побудови на основі Gradle;
- швидкий та багатофункціональний емулятор;
- єдине середовище, де можна розробляти додатки для всіх версій Android;
- шаблони коду та інтеграція GitHub, щоб допомогти вам створити загальні функції програми та імпортувати зразок коду;
- широкі інструменти та рамки тестування;
- інструменти для маніпуляцій для отримання продуктивності, зручності використання, сумісності версій та інших проблем;
- підтримка C ++ та NDK;
- вбудована підтримка Google Cloud Platform, що спрощує інтеграцію Google Cloud Messaging та App Engine.

Для підтримки розробки програм в операційній системі Android Android Studio використовує систему збірки на основі Gradle, емулятор, шаблони коду та інтеграцію з Github. Кожен проект в Android Studio має одну або кілька модальностей з вихідним кодом і файлами ресурсів. Ці модальності включають модулі програм Android, модулі бібліотеки та модулі Google App Engine.

Android Studio використовує функцію Instant Push для надсилання змін коду та ресурсів у запущену програму. Редактор коду допомагає розробнику писати код і пропонує доповнення, заломлення та аналіз коду. Програми, створені в Android Studio, потім компілюються у формат APK для надсилання в Google Play Store.

Програмне забезпечення було вперше анонсовано на конференції Google I/O у травні 2013 року, а перша стабільна збірка була випущена в грудні 2014 року. Android Studio доступна для настільних платформ Mac, Windows і Linux.

Він замінив Eclipse Android Development Tools (ADT) як основну IDE для розробки додатків Android. Android Studio і Software Development Kit можна завантажити безпосередньо з Google.

Структура проекту. Кожен проект в Android Studio містить один або кілька модулів з файлами вихідного коду та файлами ресурсів.

Типи модулів включають:

- Модулі програми для Android
- Бібліотечні модулі
- Модулі Google App Engine

manifests: містить файл AndroidManifest.xml.

java: Містить файли вихідного коду Java, включаючи тестовий код JUnit.

res: Містить усі некодові ресурси, такі як макети XML, рядки інтерфейсу користувача та растрові зображення. Структура проекту Android на диску відрізняється від цього сплощеного представлення.

3.2 Механізм вилучення, обробки і аналізу даних

3.2.1 Загальний опис алгоритмів пошуку загроз

Поширеність шкідливих програм на наших мобільних пристроях можна виявити за допомогою різних типів аналізу, таких як статичний, динамічний та гібридний підходи. У статичному підході ми збираємо набір додатків і виявляємо його на наявність загроз. У динамічному підході ми перевіряємо наявність шкідливих файлів, коли він працює на вашій платформі Android. Гібрид має поєднані риси обох підходів. Метод виявлення зловмисного програмного забезпечення в цій статті здійснюється за допомогою статичного підходу. Дослідження показують, що кількість шкідливих програм велике, і їх важко ідентифікувати. Причиною цього є методи ухилення, які використовуються для приховування шкідливого корисного навантаження, як правило, у нешкідливих програмах, і ця техніка забезпечує функціональність, яку хоче користувач. Процедуру сканування на основі сигнатур легко обійти, використовуючи

поліморфні методи та шифруючи шкідливе корисне навантаження. Завдяки цьому розкриття програм, класифікація зразків і створення тесту на нові загрози займає більше часу.

Ефективний, надійний і масштабований модуль розпізнавання зловмисного програмного забезпечення є ключовим компонентом кожного продукту безпеки. Модулі розпізнавання зловмисного програмного забезпечення вирішують, чи є об'єкт загрозою, на основі даних, які вони зібрали про нього.

Ці дані можуть бути зібрані на різних етапах:

- Дані фази перед виконанням – це все, що ви може розповісти система про файл, не виконуючи його. Це може включати описи форматів виконуваних файлів, описи коду, статистику двійкових даних, текстові рядки та інформацію, отриману за допомогою емуляції коду, та інші подібні дані.
- Дані фази після виконання передають інформацію про поведінку або подій, викликані активністю процесу в системі.

На початку кібер-ери кількість загроз зловмисного програмного забезпечення була відносно низькою, і для виявлення загроз часто було достатньо простих вручну створених правил попереднього виконання. Швидке зростання Інтернету та подальше зростання кількості шкідливих програм означали, що створені вручну правила виявлення більше не практичні – і потрібні були нові передові технології захисту. Компанії, що займаються боротьбою зі зловмисним програмним забезпеченням, звернулися до вивчення зловмисного програмного забезпечення, області інформатики, яка була успішно використана для розпізнавання зображень, пошуку та прийняття рішень, щоб покращити виявлення та класифікацію шкідливих програм. Сьогодні машинне навчання покращує виявлення зловмисного програмного забезпечення, використовуючи різні типи даних про хост, мережу та хмарні компоненти захисту від шкідливих програм.[8]

Зростаюча кількість проблем, з якими ми стикаємося сьогодні, призвела до розробки комплексного підходу для виявлення шкідливих програм для

платформи Android. Тому, у своїй роботі, вирішив обрати метод виявлення шкідливих програм Android за допомогою паралельних класифікаторів машинного навчання, які використовують контрольовані алгоритми. Використовуючи різноманітні алгоритми машинного навчання, точність механізму виявлення підвищується. Усі алгоритми, що використовуються тут, контролюються, і це дерева рішень, проста логістика, Naive Bayes.

3.2.2 Структура Android застосунків

Програми Android створені з використанням чотирьох різних компонентів

- Activities
- Services
- Broadcast receivers
- Content providers

Усі дії GUI виконуються за допомогою компонента ACTIVITY, і, як випливає з назви, фонові приймачі працюють у фоновому режимі, коли запущені програми, а дані надаються через інтерфейс постачальниками контенту. Діяльність є обов'язковими компонентами, тоді як решта необов'язкова. Програми для Android зазвичай пишуться на Java. Вони мають розширення. Apk. Зазвичай збираються в один файл разом з даними та ресурсами. APK складається з різних компонентів, як-от

- XML-файл з описом програми
- Файл classes.dex, який є виконуваним
- Каталог /res для ресурсів
- Каталог /lib для скомпільованого коду
- /meta для зберігання ресурсів та сертифікатів програми
- Ресурси. arsc, який є скомпільованим файлом ресурсу.

3.2.3 Виявлення потенційного загрозливого ПЗ за ключовими словами

Підхід виявлення шкідливого програмного забезпечення, який використовується в цій роботі, є статичним, як описано раніше. Функції програм витягуються з файлів APK, щоб перевірити, чи є вони шкідливими чи ні. Ми

використовуємо композиційну модель класифікації, щоб прийняти рішення. За допомогою спеціального інструменту аналізу APK, написаного на Java, таблиця вилучення функцій для порівняння створюється з базою даних шкідливих і небезпечних програм.

Категорії функцій для виявлення:

- функції, пов’язані з API;
- дозволи програми;
- стандартні Android фреймворки.

Функції, пов’язані з API, можна отримати з файлу classes.dex, оскільки вони містять ключові слова, які потрібно перевірити, а також використовувати для збільшення функцій програми. Необхідні дозволи програмі можна отримати з файлу маніфесту, де вони оголошуються. Linux команди також можуть служити ключовими словами.

В Табл. 3.1 наведено ключові слова-мітки, наявність яких збільшує ступінь ймовірної загрози.

Таблиця 3.1

Перелік ключових слів для подальшого розпізнання загрозливого ПЗ

ТИП	КЛЮЧОВІ СЛОВА
API ФУНКЦІЇ	abortBroadcast; getDeviceId; getSubscriberId; getCallState; getSimSerialNumber; getLineNumber; getSimCountryIso; getNetworkOperatory getSimOperator; getPackageManager; Runtime.exec(); android.provider.Contacts; android.provider.ContactsContract; HttpPost_init; HttpGet_init; HttpUriReguest; SMSReceiver; bindService; onActivityResult; Secret Key; KeySpec; FindClass; createSubprocess; Ljavax_crypto_Cipher; Ljavax_crypto_spec_Secret; DexClassLoader; sendMultipartTextMessage; Ljava_net_URLDecoder; native; System.loadLibrary; reflectgetClass; getMethod; registerReceiver; intent.action.BOOT_COMPLETED; intent.action.RUN
КОМАНДИ	mount; remount; chmod; chows; /res; /system/bin; /system/bin/sh; /system/app; .jar; .apk; pmsetInstallLocation; pminstall; GET META_DATA; GET_RECEIVERS; GET SERVICES; GET SIGNATURES; GET PERMISSIONS
ДОЗВОЛИ	ACCESS_COARSE_LOCATION; ACCESS_FINE_LOCATION; WRITE_SMS; SENDS_SMS; WRITE_CALL LOG; WRITE_APN_SETTINGS; BROADCAST_SMS; RECENE_BOOT_COMPLETED; RECEIVE_MMS; RECEIVE_SMS; RECEIVE_WAP_PUSH; RECORD_AUDIO; CALL_PHONE; WRITE_ERTERNAL CHANGE_WIFI_STATE; CLEAR APP_CACHE; INSTALL PACKAGES; INTERNET; CAMERA ;CHANGE_CONFIGURATION; CHANGE_NETWORK_STATE 1

Основну увагу потрібно звертати на дозволи, які вимагає додаток. Користувачі самі не усвідомлюють доступ до яких функцій надають застосунку.

3.2.4 Моделі машинного навчання для підходу паралельної класифікації

Алгоритм машинного навчання відіграє важливу роль у розробці інтелектуальних систем у різних областях. У методах виявлення, машинне навчання є кращим як для мобільної, так і для комп'ютерної платформи для виявлення шкідливих програм. Як було зазначено раніше, у системі моніторингу загроз інформаційним ресурсам мобільних пристроїв використовуються контрольовані алгоритми з метою виявлення та побудови набору даних шляхом вилучення з вищезазначених процедур.

Використовується підхід паралельного машинного навчання, оскільки він використовує потужність кількох алгоритмів в одному блоці. Отже, складена модель побудована на основі алгоритмів на основі функцій, дерев, імовірнісних і двох правил, заснованих на техніці суперкомп'ютера.

ДЕРЕВА РІШЕНЬ: це алгоритм, заснований на принципі «розділяй і володарюй». Як правило, це послідовна модель, яка логічно поєднує послідовність простих тестів, у даному випадку це порівняння функцій між набором даних і програмою, що підлягає тестуванню. Це більш-менш схоже на блок-схему з міткою рішення в кожному вузлі, і результат, отриманий під час кожної перевірки, додатково перевіряється.

ПРОСТА ЛОГІСТИКА: адитивна логістична регресія, що використовує прості функції регресії як основу, генерується за допомогою цього алгоритму. Алгоритм намагається знайти функцію, яка найкраще підходить для навчального набору даних. Класифікатори простої логістики займають більше часу для навчання, але швидко класифікуються.

НАІВНИЙ БАЙС: Цей алгоритм добре відомий своїм досвідом у класифікації документів та фільтрації спаму. У порівнянні з найскладнішим методом він надзвичайно швидкий.

МЕТОД ЧАСТИН: Цей алгоритм більш-менш схожий на метод «відокремлюй і перемагай». Функції з протестованих програм порівнюються з

кожним правилом, і програма призначається до першої відповідної категорії. В кінці кожної ітерації формується дерево, і найкращий листок з нього вважається правилом.

RIDOR: Цей алгоритм генерує правило за замовчуванням з найменшими можливими помилками, а також з усіма можливими винятками для цього правила.[9]

3.3 Алгоритмізація та програмування програмних модулів

3.3.1 Створення сховища даних

Щоб створити нову базу даних на мові SQL потрібно використати команду CREATE SCHEMA. Створення БД для даної системи відбувається за команди зображеної на рис. 6.

```
CREATE DATABASE [Scan_db]
```

Рис. 6 Запит на створення бази даних

Для створення таблиць програмним способом використовують оператор CREATE TABLE. Для цього потрібно вказати наступні дані:

- ім'я таблиці, яке вказується після ключового слова CREATE TABLE;
- імена та визначення стовпців таблиці, відокремлені комами.

Приклад створення таблиці для БД даного проекту наведено на рис. 7:

```
CREATE TABLE [dbo].[scannin_fact](
    [id_user] [int] NOT NULL,
    [id_file] [int] NOT NULL,
    [id_threat] [int] NOT NULL,
    [id_date] [datetime] NOT NULL,
    [result] [bit] NOT NULL,
    [threats_count] [int] NOT NULL,
    [carantine_files_count] [int] NOT NULL,
    [exception_files_count] [int] NOT NULL,
CONSTRAINT [PK_scannin_fact] PRIMARY KEY CLUSTERED
```

Рис. 7 Запит на створення таблиці «scannin»

Після виконання запиту створилася таблиця із наступною структурою (рис. 8)

Имя столбца	Тип данных	Разрешить знач...
id_user	int	☐
id_file	int	☐
id_threat	int	☐
id_date	datetime	☐
result	bit	☐
threats_count	int	☐
carantine_files_count	int	☐
exception_files_count	int	☐

Рис. 8 Структура таблиці «scannin»

SQL – запити на створення інших таблиць знаходяться у додатку А.

3.3.2 Реалізація інтерфейсу користувача.

Для реалізації інтерфейсу користувача було використано графічний редактор Adobe Photoshop CS6 та XML. Для зручності роботи користувача з програмою був обран макет додатку Navigation Drawer. Дизайн програми був розроблений за методико Material design. Material design - це мова візуальних образів, яку не так давно створила корпорація Google для уніфікації інтерфейсів всіх її продуктів і сервісів. Брендбук, який включає в себе всі елементи даного напрямку в дизайні, постійно розвивається і доповнюється, при цьому зберігаючи фундаментальні основи незмінними.

Стратегічне бачення компанії полягає в створенні нового користувацького досвіду і проникнення сервісів в усі аспекти життєдіяльності користувача. В цьому і полягає ідея єдиного інтерфейсу - об'єднати весь різноманітний і навіть розсолоту сервіс в єдиному ключі, щоб створити цілісний призначений для користувача досвід. Поверхні і контури елементів в даному напрямку дизайну створюють візуальні образи і сигнали, які передають підказки і допомагають інтуїтивно орієнтуватися, як якщо б це відбувалося в реальному світі. Стандарт XML визначає набір базових лексичних та синтаксичних правил

для побудови мови описання інформації шляхом застосування простих тегів. Він був створений для структурування, зберігання і передачі інформації.

Використання знайомих тактильних характеристик і реалістичне освітлення допомагають користувачеві візуально відокремити головні об'єкти від другорядних, зрозуміти ставлення об'єкта до його оточенню і визначити його призначення. Material design грунтується і на принципах друкованого дизайну. І не тільки для краси, а й для розстановки акцентів і фокусування уваги користувача на потрібному елементі, для спрощення навігації серед ієрархії конструкцій інтерфейсу, для інтуїтивної передачі їх сенсу.[10]

3.3.3 Забезпечення інтерфейсу з базою даних.

Зі сторони програми для взаємодії з серверною частиною використовується бібліотека Volley. Volley це бібліотека, яка робить мережеві додатки для Android простіше і, найголовніше, швидше. Вона управляє обробкою і кешуванням мережевих запитів, і це заощаджує дорогоцінний час розробників від написання одного і того ж коду мережевого запиту / зчитування з кеша знову і знову. Volley автоматично складає всі мережеві запити Volley буде приймати на себе всі мережеві запити вашого застосування виконувати їх для вилучення відповіді або зображення з веб-сайтів. Бібліотека забезпечує потужне API для скасування запиту. Можна скасувати один запит або встановити кілька запитів для скасування.[11]

4 РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ

4.1 Побудова звітності в середовищі BI

Для побудови звітів було обрано середовище MS Power BI.

Power BI - досить потужна і при цьому безкоштовна BI-платформа. Microsoft вкладає багато коштів у розвиток цього продукту, в зв'язку з чим часто виходять оновлення, що розширюють її можливості [12].

Power BI – це аналітичне середовище (комплекс програм і онлайн сервісів), яке дає можливість:

- легкого підключення до будь-вивантаженні інформації з різних джерел, об'єднання і приведення цієї інформації в єдину стандартизовану модель даних (єдиний інформаційний колодязь)

Результат підключення до сховища даних зображено на Рис. 9

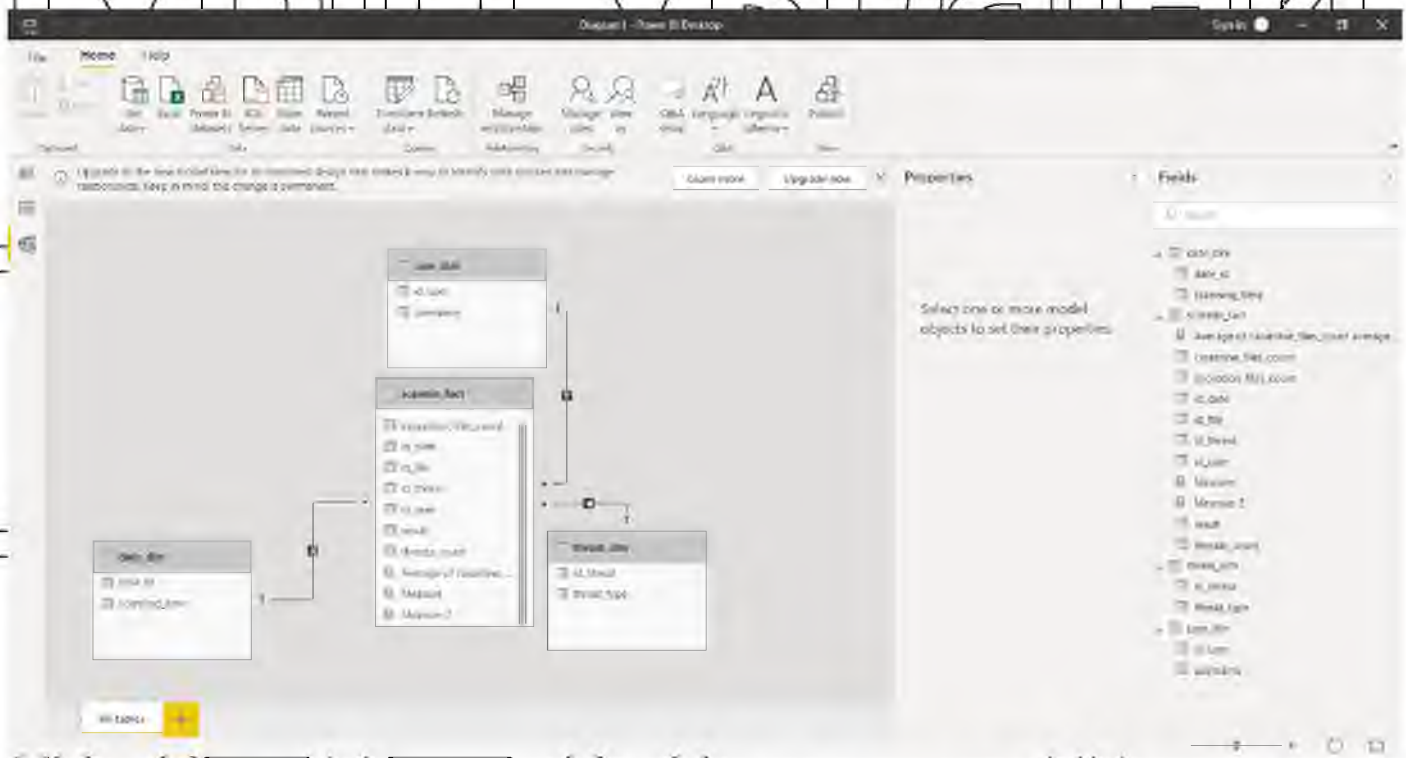

Рисунок 9.Підключення сховища даних до Power BI

- обчислення необхідних параметрів і KPI на основі цих об'єднаних даних

- побудови візуальних графіків, звітів

В даному курсовому проекті було створено 3 звіти у вигляді графіків та діаграм. Було створено звіт, де показано кількість сканувань та виявлених загроз по місяцях (Рис. 10) та кількості загроз від їх типу (Рис. 11).

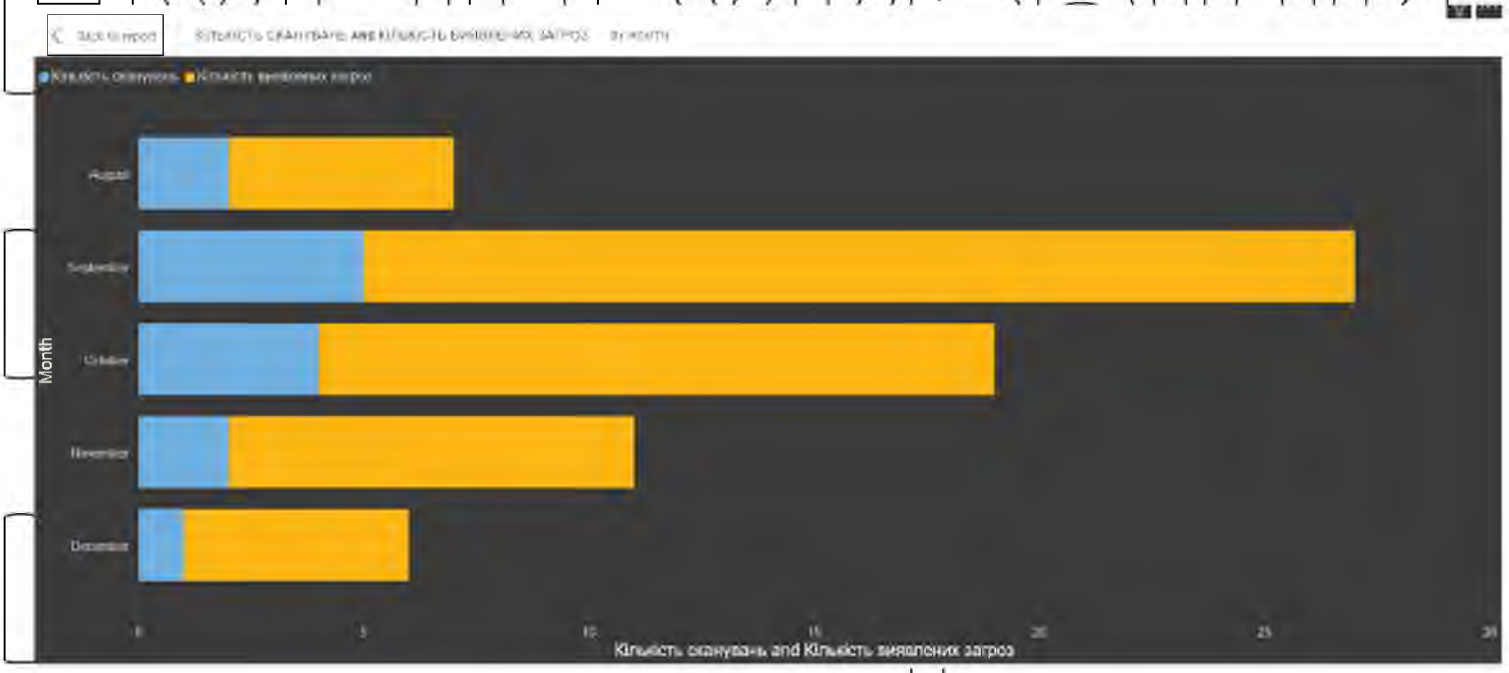

Рис. 10 Звіт " Кількість сканувань та виявлених загроз за період часу вересень-грудень 2021р."

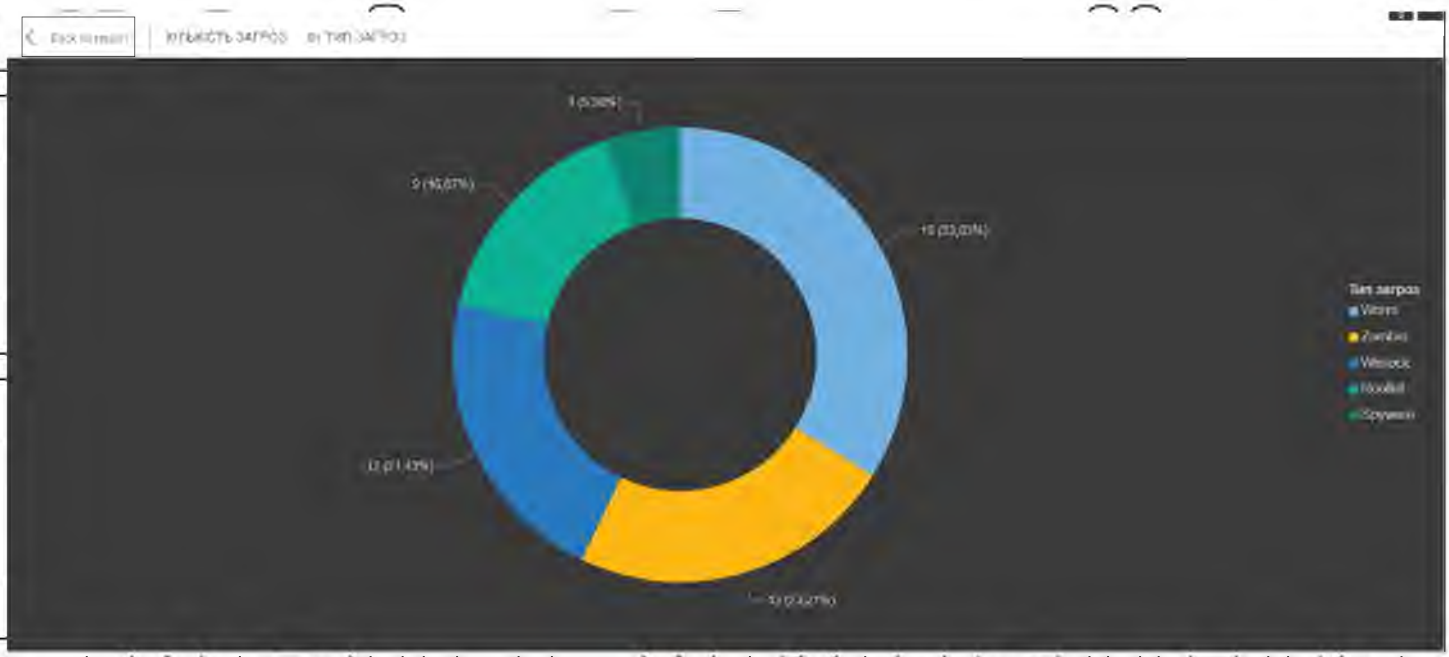

Рис. 11 Звіт " Залежність кількості загроз від типу"

Проаналізувавши діаграму, видно, що найбільша частка загроз припадає на загрозу типу «Хробак», після нього зі, майже однаковою, часткою розташувалися загрози типу «Зомбі» та «Віруси-блокувальники». Найменш розповсюдженим був вірус типу «Шпигун».

На наступному звіті (Рис.). можна побачити кількість файлів на карантині та файлів-виключень кожного користувача, тобто оцінити ступінь зараженості пристрою окремого користувача.

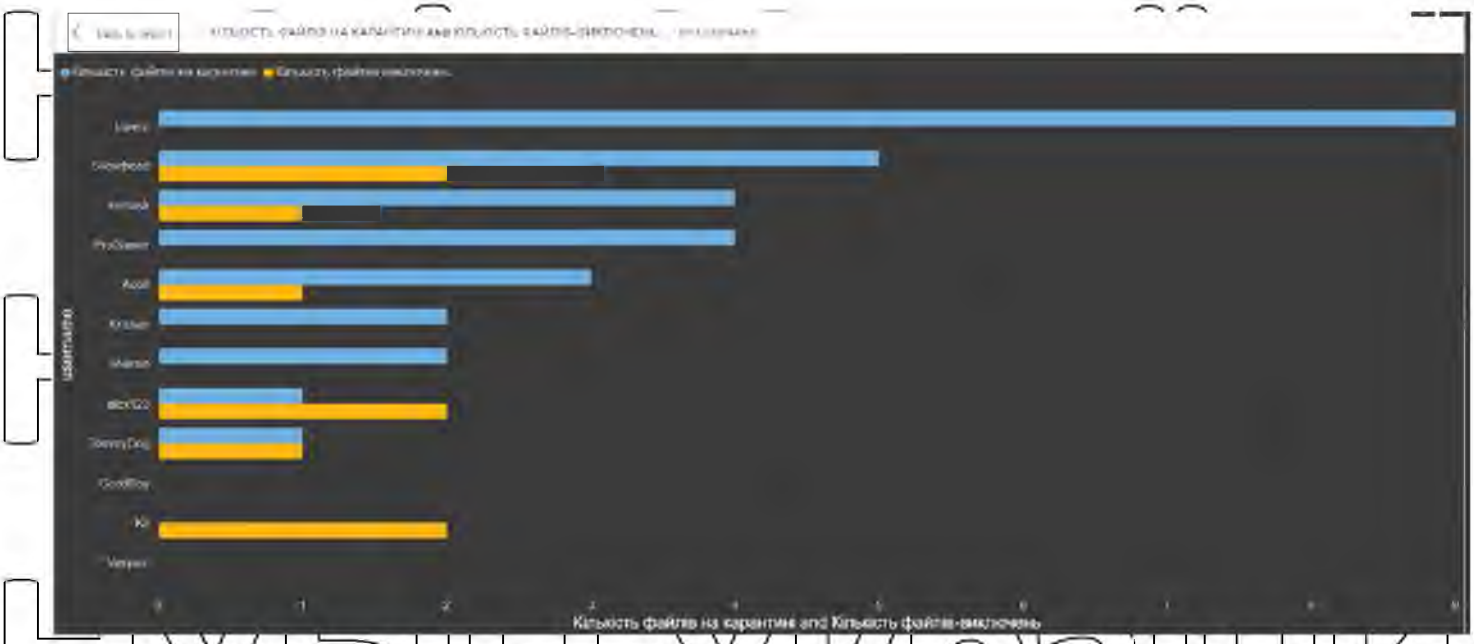

Рис. 12 Кількість файлів на карантині та файлів-виключень кожного користувача за період часу вересень – грудень 2021р.

4.3 Вимоги до апаратного та програмного забезпечення

4.3.1 Вимоги до апаратного забезпечення. Для коректного виконання програми рекомендується наступне **апаратне забезпечення:**

- Процесор – мінімальна частота 1.5 ГГц;
- Кількість ядер процесору – мінімальна 2;
- ОЗУ – мінімум 2 ГБ;
- Вбудована пам’ять – мінімум 3 ГБ вільної;
- Інтернет з’єднання – швидкість мережі від 20 Мбіт/с.

4.3.2 Вимоги до програмного забезпечення. Для інсталяції та запуску додатку необхідне наступне **програмне забезпечення**:

- ОС Android – мінімальна версія 4.4;
- SDK – мінімальна версія 19.

ВИСНОВКИ

В результаті виконання роботи розроблене сховище даних, програмне забезпечення та моделювання для забезпечення моніторингу загроз інформаційним ресурсам мобільних пристроїв.

У першому розділі було проведено попередній аналіз предметної області, сформовано ціль, мету та завдання роботи. Проведено аналіз наявних аналогів застосунків моніторингу загроз інформаційним ресурсам мобільних пристроїв. Визначені основні типи шкідливого програмного забезпечення та можливі наслідки спричинені ними.

У другому розділі визначено користувачів та архітектуру системи. Описано основні вузли системи та розроблено сховище даних. Побудовано логічну схему сховища даних, а також досліджено основні поняття з напрямку OLAP-технології.

У третьому розділі описано вибір методів та засобів для реалізації інформаційного забезпечення системи. Для створення сховища даних проекту обрано систему управління базою даних – Microsoft SQL Server. Описано структуру Android додатку та визначено основні методи розпізнання загрозливого ПЗ.

Четвертий розділ присвячено розробці системи аналізу. Було проведено опис середовища розробки Microsoft BI SQL Server, а також Power BI. Після цього було проведено аналіз існуючих даних та побудовано звіти для аналізу кількісної статистики.

Отримана модель може бути використана для створення програмного продукту та зменшити вірогідність стати жертвою зловмисників. Отже, можна зробити висновок, що мета та завдання курсової роботи були виконані.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1 The Best Android Antivirus apps [Електронний ресурс]. – 2021 - Режим доступу до ресурсу: https://www.pcmag.com/picks/the-best-android-antivirus-apps

2 The 11 most common types of malware [Електронний ресурс]: – 2021 - Режим доступу до ресурсу: https://www.crowdstrike.com/cybersecurity-101/malware/types-of-malware/

3 What is Unified Modeling Language (UML)? [Електронний ресурс]: – 2016 – Режим доступу до ресурсу: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/

4 OLAP системи [Електронний ресурс]: // KPMS - Режим доступу до ресурсу: https://www.kpms.ru/Automatization/OLAP_system.html

5 Що таке сховище даних? [Електронний ресурс]: // Oracle - Режим доступу до ресурсу: https://www.oracle.com/ru/database/what-is-a-datawarehouse/

6 Поняття архітектури системи [Електронний ресурс]: – 2020 - Режим доступу до ресурсу: http://lib.mdpu.org.ua/e-book/vstup/L6.htm#L6

7 Business Intelligence: What It Is, How It Works [Електронний ресурс]: – 2020 - Режим доступу до ресурсу: https://www.tableau.com/learn/articles/business-intelligence

8 Kaspersky-Lab-Whitepaper-Machine-Learning [Електронний ресурс]: – 2019 – Режим доступу до ресурсу: https://media.kaspersky.com/en/enterprise-security/Kaspersky-Lab-Whitepaper-Machine-Learning.pdf

9 Detection Of Malware Applications In Mobile Devices [Електронний ресурс]: – 2017 - Режим доступу до ресурсу: https://www.researchgate.net/publication/321873913_Detection_Of_Malware_Applications_In_Mobile_Devices_Using_Supervised_Machine_Learning_Algorithms

10 Что такое material design? [Електронний ресурс]: – 2019 - Режим доступу до ресурсу: https://darksiteofmarketing.com/stati/chto-takoe-material-design.html

11 Простой пример использования библиотеки Volley [Електронний ресурс]: – 2013 - Режим доступу до ресурсу: https://habr.com/ru/post/188360/

12 BI - бізнес-аналітика [Електронний ресурс]: // It.ua - Режим доступу до ресурсу: https://www.it.ua/ru/knowledge-base/technology-innovation/business-intelligence-bi

13 Що таке тестування програмного забезпечення? [Електронний ресурс]: – 2017 - Режим доступу до ресурсу: https://qalight.com.ua/baza-znaniy/chto-takoe-testirovanie-programmnogo-obespecheniya/

14 Клієнт-серверна архітектура та ролі серверів. [Електронний ресурс]: – 2017 - Режим доступу до ресурсу: https://medium.com/@IvanZmerzlyi/

15 Top Ten Advantages of Android [Електронний ресурс]: – 2019 - Режим доступу до ресурсу: https://thisisglance.com/top-ten-advantages-of-android/

16 What is a Data Warehouse and Why Does It Matter To Your Business? [Електронний ресурс] – 2019 - Режим доступу до ресурсу: https://www.talend.com/resources/what-is-data-warehouse/

17 The Advantages and Disadvantages of Microsoft SQL Server [Електронний ресурс] – 2020 - Режим доступу до ресурсу: https://www.rothmobot.com/the-advantages-and-disadvantages-of-microsoft-sql-server/

18 Business Intelligence: What It Is, How It Works, Its Importance, Examples, & Tools [Електронний ресурс]: – 2020 - Режим доступу до ресурсу: https://www.tableau.com/learn/articles/business-intelligence

ДОДАТОК А

SQL ЗАПИТИ СТВОРЕННЯ ТАБЛИЦЬ СХОВИЩА ДАНИХ

```
USE [Scan_db]
GO

/****** Object:  Table [dbo].[scannin_fact]    Script Date: 29.11.2021 23:48:03 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[scannin_fact](
	[id_user] [int] NOT NULL,
	[id_file] [int] NOT NULL,
	[id_threat] [int] NOT NULL,
	[id_date] [datetime] NOT NULL,
	[result] [bit] NOT NULL,
	[threats_count] [int] NOT NULL,
	[carantine_files_count] [int] NOT NULL,
	[exception_files_count] [int] NOT NULL,
 CONSTRAINT [PK_scannin_fact] PRIMARY KEY CLUSTERED
(
	[id_user] ASC,
	[id_file] ASC,
	[id_threat] ASC,
	[id_date] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
 = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO

ALTER TABLE [dbo].[scannin_fact]  WITH CHECK ADD  CONSTRAINT [FK_scannin_fact_date_dim]
FOREIGN KEY([id_date])
REFERENCES [dbo].[date_dim] ([date_id])
GO

ALTER TABLE [dbo].[scannin_fact] CHECK CONSTRAINT [FK_scannin_fact_date_dim]
GO

ALTER TABLE [dbo].[scannin_fact]  WITH CHECK ADD  CONSTRAINT [FK_scannin_fact_file_dim]
FOREIGN KEY([id_file])
REFERENCES [dbo].[file_dim] ([id_file])
GO

ALTER TABLE [dbo].[scannin_fact] CHECK CONSTRAINT [FK_scannin_fact_file_dim]
GO

ALTER TABLE [dbo].[scannin_fact]  WITH CHECK ADD  CONSTRAINT [FK_scannin_fact_threat_dim]
FOREIGN KEY([id_threat])
REFERENCES [dbo].[threat_dim] ([id_threat])
GO

ALTER TABLE [dbo].[scannin_fact] CHECK CONSTRAINT [FK_scannin_fact_threat_dim]
GO

ALTER TABLE [dbo].[scannin_fact]  WITH CHECK ADD  CONSTRAINT [FK_scannin_fact_user_dim]
FOREIGN KEY([id_user])
REFERENCES [dbo].[user_dim] ([id_user])
GO

ALTER TABLE [dbo].[scannin_fact] CHECK CONSTRAINT [FK_scannin_fact_user_dim]
GO
```

Сторінка 2

```
USE [Scan_db]
GO

/****** Object:  Table [dbo].[file_dim]    Script Date: 29.11.2021 23:48:53 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[file_dim](
	[id_file] [int] NOT NULL,
	[File_name] [nchar](10) NULL,
 CONSTRAINT [PK_File_dim] PRIMARY KEY CLUSTERED
(
	[id_file] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO

USE [Scan_db]
GO

/****** Object:  Table [dbo].[date_dim]    Script Date: 29.11.2021 23:49:16 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[date_dim](
	[date_id] [datetime] NOT NULL,
	[scanning_time] [int] NOT NULL,
 CONSTRAINT [PK_date_dim] PRIMARY KEY CLUSTERED
(
	[date_id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
USE [Scan_db]
GO
/****** Object:  Table [dbo].[threat_dim]    Script Date: 29.11.2021 23:50:10 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[threat_dim](
	[id_threat] [int] NOT NULL,
	[threat_type] [nchar](10) NOT NULL,
 CONSTRAINT [PK_threat_dim] PRIMARY KEY CLUSTERED
(
	[id_threat] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
```

Сторінка 3

```
USE [Scan_db]
GO

/****** Object:  Table [dbo].[user_dim]    Script Date: 29.11.2021 23:50:24 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[user_dim](
	[id_user] [int] NOT NULL,
	[username] [nchar](15) NOT NULL,
 CONSTRAINT [PK_user_dim] PRIMARY KEY CLUSTERED
(
	[id_user] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
 = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
```

ДОДАТОК Б

ВМІСТ ОСНОВНОГО КЛАСУ ПРОГРАМИ

```
package com.example.potentialThreads;

import androidx.appcompat.app.AppCompatActivity;
import androidx.constraintlayout.widget.ConstraintLayout;
import androidx.core.app.ActivityCompat;
import android.Manifest;

import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.pm.ApplicationInfo;
import android.content.pm.PackageInfo;
import android.content.pm.PackageManager;
import android.os.Bundle;
import android.os.StrictMode;
import android.util.Log;
import android.util.TypedValue;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ListView;
import android.widget.TextView;

import java.sql.Connection;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;

public class MainActivity extends AppCompatActivity {

    private TextView textView;
    private TextView textViewD;
```

```
    private String[] requestedPermissions;
    private int  mlSelectedInd = 0;
    private ArrayList<ArrayList<String>> outer = new ArrayList<>();
    private ArrayList<String> inner = new ArrayList<>();
    private ArrayList<String> threadsArr = new ArrayList<>();

    private static String ip = "192.168.0.102";
    private static String port = "1433";
    private static String classes = "net.sourceforge.jtds.jdbc.Driver";
    private static String database = "Scan_db";
    private static String user = "test";
    private static String pass = "test";
    private static String url = "jdbc:jtds:sqlserver://"+ip+":"+port+"/"+database;


    private ListView lw;

    private Connection connection = null;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);


        ActivityCompat.requestPermissions(this,new String[]{Manifest.permission.INTERNET},
PackageManager.PERMISSION_GRANTED);

        lw = (ListView) findViewById(R.id.listView1);

        StrictMode.ThreadPolicy policy = new
StrictMode.ThreadPolicy.Builder().permitAll().build();
        StrictMode.setThreadPolicy(policy);
```

```
            /*try {
                Class.forName(classes);

                connection = DriverManager.getConnection(url, user, pass);

                textView.setText("SUCCESS");
            } catch (ClassNotFoundException | SQLException e) {
                e.printStackTrace();
                textView.setText("ERROR");
            }*/

            lw.setOnItemClickListener(new AdapterView.OnItemClickListener() {
                @Override
                public void onItemClick(AdapterView<?> parent, View view, int position, long id)
        {
                    mlSelectedInd = position;
                    myCustDialog();

                }
            });
        }
    //    public void runDialog(){

    //  }
        public void myCustDialog(){
            int countThread = getThreadCounts(outer.get(mlSelectedInd));

            TextView dialogTitle = new TextView(getApplicationContext());
            if (countThread!=0){
                dialogTitle.setText("\t\t\tСписок дозволів\nКількість підозрюваних дозволів:
"+countThread+"\n"+threadsArr);
```

```
        }else{
            dialogTitle.setText("\t\t\tСписок дозволів\nПідозрюваних дозволів не виявлено");
        }

        dialogTitle.setTextColor(getResources().getColor(R.color.cardview_dark_background));
        dialogTitle.setTextSize(TypedValue.COMPLEX_UNIT_SP,20);

        final String[] permArray = outer.get(mlSelectedInd).toArray(new String[0]);


        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setPositiveButton("Закрити", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
            }

        })
                //.setTitle("Список дозволів Кількість підозрюваних дозволів:
"+countThread+" ("+threadsArr+")")
                .setCustomTitle(dialogTitle)
        .setItems(permArray, new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                dialog.dismiss();
            }
        });
        ConstraintLayout cl = (ConstraintLayout)
getLayoutInflater().inflate(R.layout.dial_lay, null);

        builder.setView(cl);
        builder.show();
    }
    public void usegetPackageInfo(View view){
        //
        final ArrayList<String> al = new ArrayList<>();
        // final ListView lw = (ListView) findViewById(R.id.listView1);
        lw = (ListView) findViewById(R.id.listView1);
        final ArrayAdapter<String> adapter;
```

```
        adapter = new ArrayAdapter<>(this,
                android.R.layout.simple_list_item_1, al);
        lw.setAdapter(adapter);

        PackageManager p = lw.getContext().getPackageManager();
        final List<PackageInfo> appinstall =
                p.getInstalledPackages(PackageManager.GET_PERMISSIONS |
                        PackageManager.GET_PROVIDERS);
        //final TextView tw = (TextView) findViewById(R.id.textView);
        Iterator it = appinstall.iterator();
        while (it.hasNext()) {
            PackageInfo rf = (PackageInfo) it.next();
            al.add(0, rf.toString());
            adapter.notifyDataSetChanged();
            //tw.append(rf.toString()+ "\n");
        }
    }
    public int getThreadCounts(List<String> arraylist) {
        threadsArr.clear();
        int countThread = 0;
        for (String str : arraylist) {
            if (str.toUpperCase().contains("ACCESS_COARSE_LOCATION") ||
str.toUpperCase().contains("ACCESS_FINE_ LOCATION") ||
str.toUpperCase().contains("WRITE_SMS")
                    || str.toUpperCase().contains("WRITE_SMS") ||
str.toUpperCase().contains("WRITE_CALL LOG") ||
str.toUpperCase().contains("WRITE_APN_SETTINGS")
                    || str.toUpperCase().contains("BROADCAST_SMS") ||
str.toUpperCase().contains("RECENE_BOOT_COMPLETED") ||
str.toUpperCase().contains("RECEIVE_MMS")
                    || str.toUpperCase().contains("RECEIVE_SMS") ||
str.toUpperCase().contains("RECEIVE_WAP_PUSH") || str.toUpperCase().contains("RECORD_AUDIO")
                    || str.toUpperCase().contains("CALL_PHONE") ||
str.toUpperCase().contains("WRITE_ERTERNAL CHANGE_WIFI_STATE") ||
str.toUpperCase().contains("CLEAR APP_CACHE")
                    || str.toUpperCase().contains("CHANGE_CONFIGURATION") ||
str.toUpperCase().contains("CHANGE_NETWORK_STATE 1")
                    || str.toUpperCase().contains("INSTALL_PACKAGES") ||
str.toUpperCase().contains("INTERNET") || str.toUpperCase().contains("CAMERA")) {
```

```
                threadsArr.add(str);
                countThread++;
            }
        }
        return countThread;
    }
    public void getPackagePrmiss(View view) {
        Button button = findViewById(R.id.button);
        button.setEnabled(false);
        final ArrayList<String> al = new ArrayList<>();
        lw = (ListView) findViewById(R.id.listView1);
        ArrayAdapter<String> adapter = new ArrayAdapter<>(this,
                android.R.layout.simple_list_item_1, al);
        lw.setAdapter(adapter);
        PackageManager pm = getPackageManager();
        List<ApplicationInfo> packages =
pm.getInstalledApplications(PackageManager.GET_META_DATA);
        for (ApplicationInfo applicationInfo : packages) {
            Log.d("test", "App: " + applicationInfo.name + " Package: " +
applicationInfo.packageName);
            if (applicationInfo.name != null){
            al.add(0, applicationInfo.name );
            }
            else {
                al.add(0, applicationInfo.packageName );
            }
            try {
                PackageInfo packageInfo = pm.getPackageInfo(applicationInfo.packageName,
PackageManager.GET_PERMISSIONS);
                //Get Permissions
                //String[] requestedPermissions = packageInfo.requestedPermissions;
                requestedPermissions = packageInfo.requestedPermissions;
                if(requestedPermissions != null) {
                    for (int i = 0; i < requestedPermissions.length; i++) {
                        Log.d("test permiss: ", requestedPermissions[i]);
                        inner.add(requestedPermissions[i]);
                    }
```

```
                inner = new ArrayList<String>(inner);
                outer.add(inner);
                inner = new ArrayList<>();
            }
        } catch (PackageManager.NameNotFoundException e) {
            e.printStackTrace();
        }
    }
    Log.d("INNER CONTAIN: ", inner.toString());
    Log.d("OUTER CONTAIN: ", outer.toString());
  }
}
```