Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ ТЕСТУВАННЯ НА ПРОНИКНЕННЯ МОБІЛЬНОГО ЗАСТОСУНКУ»

Виконав: студент 2 курсу,
групи 1БС-22м
спеціальності 125 Кібербезпека
_____Михайло ВОРОЖБИТ

Керівник: к. т. н., доцент каф. ЗІ
_____Леонід КУПЕРШТЕЙН
«_____»_____2023 р.

Опонент: к. т. н. доцент каф. ПЗ
_____Олена КОВАЛЕНКО
«_____»_____2023 р.

Допущено до захисту
Завідувач кафедри ЗІ
д. т. н., проф.
_____Володимир ЛУЖЕЦЬКИЙ
«_____»_____2023 р.

Вінниця ВНТУ – 2023 року

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ ТЕСТУВАННЯ НА ПРОНИКНЕННЯ МОБІЛЬНОГО ЗАСТОСУНКУ»

Виконав: студент 2 курсу,
групи 1БС-22м
спеціальності 125 Кібербезпека
_____Михайло ВОРОЖБИТ

Керівник: к. т. н., доцент каф. ЗІ
_____Леонід КУПЕРШТЕЙН
«12» 12 2023 р.

Опонент: к. т. н. доцент каф. ПЗ
_____Олена КОВАЛЕНКО
«13» 12 2023 р.

Допущено до захисту
Завідувач кафедри ЗІ
д. т. н., проф.
_____Володимир ЛУЖЕЦЬКИЙ
«14» 12 2023 р.

Вінниця ВНТУ – 2023 року

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації
Рівень вищої освіти II (магістерський)
Галузь знань – 12 «Інформаційні технології»
Спеціальність – 125 «Кібербезпека»
Освітньо-професійна програма – «Безпека інформаційних і комунікаційних систем»

ЗАТВЕРДЖУЮ
Зав. кафедри ЗІ, д. т. н., проф.
_____Володимир ЛУЖЕЦЬКИЙ
_____2023 року

ЗАВДАННЯ
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
Ворожбиту Михайлу Вікторовичу

1. Тема роботи: «Інформаційна технологія тестування на проникнення мобільного застосунку» керівник роботи: Куперштейн Леонід Михайлович, к. т. н., доцент, затверджені наказом ВНТУ від 18 вересня 2023 року №247.
2. Строк подання студентом роботи – 13 вересня 2023 року
3. Вихідні дані до роботи:
   - засоби тестування мобільних застосунків на проникнення;
   - методи тестування мобільних застосунків на проникнення.
4. Зміст текстової частини: Вступ. 1. Аналіз предметної області. 2. Розробка інформаційної технології. 3. Реалізація інформаційної технології. 4. Економічне обґрунтування. Висновки. Список використаних джерел. Застосунки.
5. Перелік графічного матеріалу: Схема процесу тестування на проникнення мобільних застосунків (плакат, А4). Схема процесів технології на тестування проникнення (плакат, А4). Схема алгоритму тестування мобільного застосунку (плакат, А4).

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації
Рівень вищої освіти II (магістерський)
Галузь знань – 12 «Інформаційні технології»
Спеціальність – 125 «Кібербезпека»
Освітньо-професійна програма – «Безпека інформаційних і комунікаційних систем»

ЗАТВЕРДЖУЮ
Зав. кафедри ЗІ, д. т. н., проф.
Володимир ЛУЖЕЦЬКИЙ
19. 09 2023 року

ЗАВДАННЯ
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
Ворожбиту Михайлу Вікторовичу

1. Тема роботи: «Інформаційна технологія тестування на проникнення мобільного застосунку» керівник роботи: Куперштейн Леонід Михайлович, к. т. н., доцент, затверджені наказом ВНТУ від 18 вересня 2023 року №247.
2. Строк подання студентом роботи – 13 вересня 2023 року
3. Вихідні дані до роботи:
   - засоби тестування мобільних застосунків на проникнення;
   - методи тестування мобільних застосунків на проникнення.
4. Зміст текстової частини: Вступ. 1. Аналіз предметної області. 2. Розробка інформаційної технології. 3. Реалізація інформаційної технології. 4. Економічне обґрунтування. Висновки. Список використаних джерел. Застосунки.
5. Перелік графічного матеріалу: Схема процесу тестування на проникнення мобільних застосунків (плакат, А4). Схема процесів технології на тестування проникнення (плакат, А4). Схема алгоритму тестування мобільного застосунку (плакат, А4).

5. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завданн явидав	завдання прийняв
1	Леонід КУПЕРШТЕЙН, доц. кафедри ЗІ		
2	Леонід КУПЕРШТЕЙН, доц. кафедри ЗІ		
3	Леонід КУПЕРШТЕЙН, доц. кафедри ЗІ		
4	Ольга РАТУШНЯК, к.т.н., доц. каф ЕВПМ		

6. Дата видачі завдання – 1 вересня 2023 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Зміст етапу	Строк виконання етапів роботи	Примітка
1	Аналіз завдання. Вступ	01.09.2023 – 10.09.2023	
2	Аналіз інформаційних джерел за напрямком магістерської кваліфікаційної роботи	10.09.2023 – 15.09.2023	
3	Науково-технічне обґрунтування	16.09.2023 – 22.09.2023	
5	Аналіз інформаційної технології	23.09.2023 – 29.09.2023	
7	Розробка інформаційної технології	17.10.2023 – 14.11.2023	
8	Тестування інформаційної технології	15.11.2023 – 17.11.2023	
9	Розробка розділу економічного обґрунтування доцільності розробки	18.11.2023 – 21.11.2023	
10	Висновки	22.11.2023 – 24.11.2023	
11	Оформлення пояснювальної записки	25.11.2023 – 29.11.2023	
12	Перевірка магістерської роботи на наявність плагіату	30.11.2023 – 02.12.2023	
13	Попередній захист та доопрацювання МКР	03.12.2023 – 11.12.2023	
14	Представлення МКР до захисту, рецензування	11.12.2023 – 14.12.2023	
16	Захист МКР	14.12.2023 – 21.12.2023	

Студент_____Михайло ВОРОЖБИТ
(Підпис)

Керівник роботи_____Леонід КУПЕРШТЕЙН
(Підпис)

5. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завданн явидав	завданн прийняв
1	Леонід КУПЕРШТЕЙН, доц. кафедри ЗІ	19.09	25.09
2	Леонід КУПЕРШТЕЙН, доц. кафедри ЗІ	19.09	27.09
3	Леонід КУПЕРШТЕЙН, доц. кафедри ЗІ	19.09	26.10
4	Ольга РАТУШНЯК, к.т.н., доц. каф ЕВПМ	19.09	17.11

6. Дата видачі завдання – 1 вересня 2023 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Зміст етапу	Строк виконання етапів роботи	Примітка
1	Аналіз завдання. Вступ	01.09.2023 – 10.09.2023	
2	Аналіз інформаційних джерел за напрямком магістерської кваліфікаційної роботи	10.09.2023 – 15.09.2023	
3	Науково-технічне обґрунтування	16.09.2023 – 22.09.2023	
5	Аналіз інформаційної технології	23.09.2023 – 29.09.2023	
7	Розробка інформаційної технології	17.10.2023 – 14.11.2023	
8	Тестування інформаційної технології	15.11.2023 – 17.11.2023	
9	Розробка розділу економічного обґрунтування доцільності розробки	18.11.2023 – 21.11.2023	
10	Висновки	22.11.2023 – 24.11.2023	
11	Оформлення пояснювальної записки	25.11.2023 – 29.11.2023	
12	Перевірка магістерської роботи на наявність плагіату	30.11.2023 – 02.12.2023	
13	Попередній захист та доопрацювання МКР	03.12.2023 – 11.12.2023	
14	Представлення МКР до захисту, рецензування	11.12.2023 – 14.12.2023	
16	Захист МКР	14.12.2023 – 21.12.2023	

Студент_____(Підпис)_____Михайло ВОРОЖБИТ

Керівник роботи_____(Підпис)_____Леонід КУПЕРШТЕЙН

АНОТАЦІЯ

УДК 004.56.5

Ворожбит М. В. Інформаційна технологія тестування на проникнення мобільного застосунку. Магістерська кваліфікаційна робота зі спеціальності 125 – Кібербезпека, освітня програма – Безпека інформаційних і комунікаційних систем. Вінниця: ВНТУ, 2023. 105 с.

Укр. мовою. Бібліогр.: 44 назв.: рис.: 26; табл.: 22.

Магістерська кваліфікаційна робота присвячена вдосконаленні технологій тестування на проникнення мобільних застосунків. В рамках роботи проведено аналіз існуючих методів та засобів тестування на проникнення мобільних застосунків.

В економічному розділі оцінено витрати на розробку інформаційної технології тестування на проникнення мобільного застосунку

Ключові слова: тестування на проникнення, операційна система андроїд, збір даних.

ABSTRACT

Vorozhbyt M. V. Information technology of mobile application penetration testing. Master's qualification work in specialty 125 – Cybersecurity, educational program – Security of information and communication systems. Vinnytsia: VNTU, 2023. 105 p.

In Ukrainian. Bibliography: 44 titles; Figs. 26; Tables: 22.

Master's thesis is devoted to the improvement of mobile application penetration testing technologies. As part of the work, an analysis of existing methods and tools for mobile application penetration testing was conducted.

The economic section estimates the costs of developing information technology for mobile application penetration testing.

Keywords: penetration testing, android operating system, data collection.

ЗМІСТ

ВСТУП

Сучасний світ інформаційних технологій постійно розвивається, пропонуючи широкий спектр мобільних застосунків, які грають важливу роль у повсякденному житті людей і сферах бізнесу. Мобільні застосунки забезпечують користувачам доступ до розмаїтої інформації, послуг та розваг, і є невід'ємною частиною цифрового середовища. Зростання популярності мобільних платформ призвело до появи нових загроз безпеці інформації, пов'язаних з використанням мобільних застосунків.

Одним із важливих завдань, пов'язаних з розробкою та використанням мобільних застосунків, є забезпечення їх безпеки та надійності. З плином часу виникає все більша потреба у захисті інформації, оскільки застосунки містять конфіденційну інформацію, особисті дані та доступ до мережевих ресурсів. Важливо пам'ятати, що кіберзлочинці постійно шукають нові способи проникнення до мобільних застосунків для крадіжки інформації або завдавання інших шкодливих наслідків.

Для цього потрібно проводити тестування на проникнення мобільних застосунків. Тестування на проникнення є необхідною складовою процесу забезпечення інформаційної безпеки. Основна мета полягає у виявленні слабкостей у системах та застосунках перед тим, як зловмисники зможуть їх використати. Мобільні застосунки, так само як будь-які інші програми, можуть мати потенційно небезпечні моменти в системі захисту, яким потребується ідентифікація й усунення. Тестування на проникнення дозволяє оцінити ефективність заходів безпеки й прийняти кроки для їх поліпшення.

Актуальність. Тестування на проникнення – один із найбільш ефективних способів покращення рівня безпеки мобільного застосунку. Тестування на проникнення для мобільних застосунків є критично важливим процесом у сфері безпеки мобільних застосунків. Він включає в себе комплексну оцінку стану безпеки мобільного застосунку шляхом імітації атак, що імітують дії зловмисників. Основною метою такого тестування є систематичне виявлення

вразливостей і слабких місць у засобах управління безпекою мобільного застосунку, що в кінцевому підсумку дає змогу отримати інформацію та рекомендації щодо їх усунення.

За останні роки мобільні застосунки стали свідками вибухового зростання та стали невід'ємною частиною нашого повсякденного життя. Однак ця всюдисущість також зробила їх основною мішенню для кіберзлочинців, які прагнуть використати слабкі місця в їхній безпеці. Ось чому тестування проникнення мобільних застосунків має таке значення [1].

Основними цілями тестування на проникнення мобільних застосунків є:

– Виявлення Вразливостей: Тестування допомагає виявити слабкі місця та уразливості в застосунку, такі як недостатньо захищені API, недоречно зберігається конфіденційна інформація, недоліки в аутентифікації та авторизації, і багато інших.

– Захист Користувачів: Проведення тестування дозволяє ідентифікувати потенційні загрози для безпеки користувачів мобільного застосунку і приймати відповідні заходи для їх захисту.

– Захист Даних: Виявлення вразливостей, які можуть призвести до витоку конфіденційної інформації або зміни даних, що зберігаються в застосунку.

– Забезпечення Відповідності: Деякі регулятори та магазини застосунків встановлюють вимоги до безпеки мобільних застосунків. "Тестування на проникнення" допомагає переконатися, що застосунок відповідає цим вимогам.

– Захист Репутації: Виявлення і вирішення проблем безпеки допомагає уникнути витоку даних та інцидентів, що можуть завдати шкоди репутації організації.

– Заощадження Коштів: Попередження інцидентів безпеки може заощадити кошти, які можуть витрачатися на відновлення та відшкодування збитків.

Результатами тестування на проникнення мобільних застосунків є звіт, який включає в себе виявлені уразливості, методи їх виправлення та рекомендації щодо підвищення безпеки застосунку.

Об'єктом дослідження є процес тестування на проникнення мобільних застосунків.

Предметом дослідження є методи та засоби тестування на проникнення мобільних застосунків.

Метою магістерської кваліфікаційної роботи є підвищення захищеності мобільного застосунку за рахунок розробки та застосування інформаційної технології тестування на проникнення.

Для досягнення мети необхідно виконати наступні завдання:

- проаналізувати проблеми безпеки мобільного застосунку;
- проаналізувати методи та засоби тестування мобільних застосунків на проникнення;
- розробити структуру інформаційної технології;
- виконати тестування програмного засобу на проникнення;
- оцінити результати тестування та надати рекомендації для покращення захисту мобільного застосунку;

Методи дослідження. Для реалізації поставлених задач були використані методи чорної скриньки та білої скриньки для тестування на проникнення програмних застосунків.

Наукова новизна. Запропонована інформаційна технологія тестування на проникнення мобільних застосунків, яка полягає в аналізі вразливостей мобільних застосунків, формуванні результату аналізу вразливостей, тестуванні мобільного застосунку та у звіті із описом результатів, які будуть надавати розуміння як більш ефективніше проводити тестування на проникнення мобільних застосунків.

Практична цінність полягає у тому, що запропоновану інформаційну технологію можна використовувати для тестування будь яких мобільних застосунків на основі ОС Android.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз безпеки операційної системи Android

Операційна система Android - це система з відкритим кодом. Android має можливість вільно змінювати, винаходити та впроваджувати власні драйвери пристроїв та функції. Операційної системи Android складається з шести різних шарів (рис 1.1).

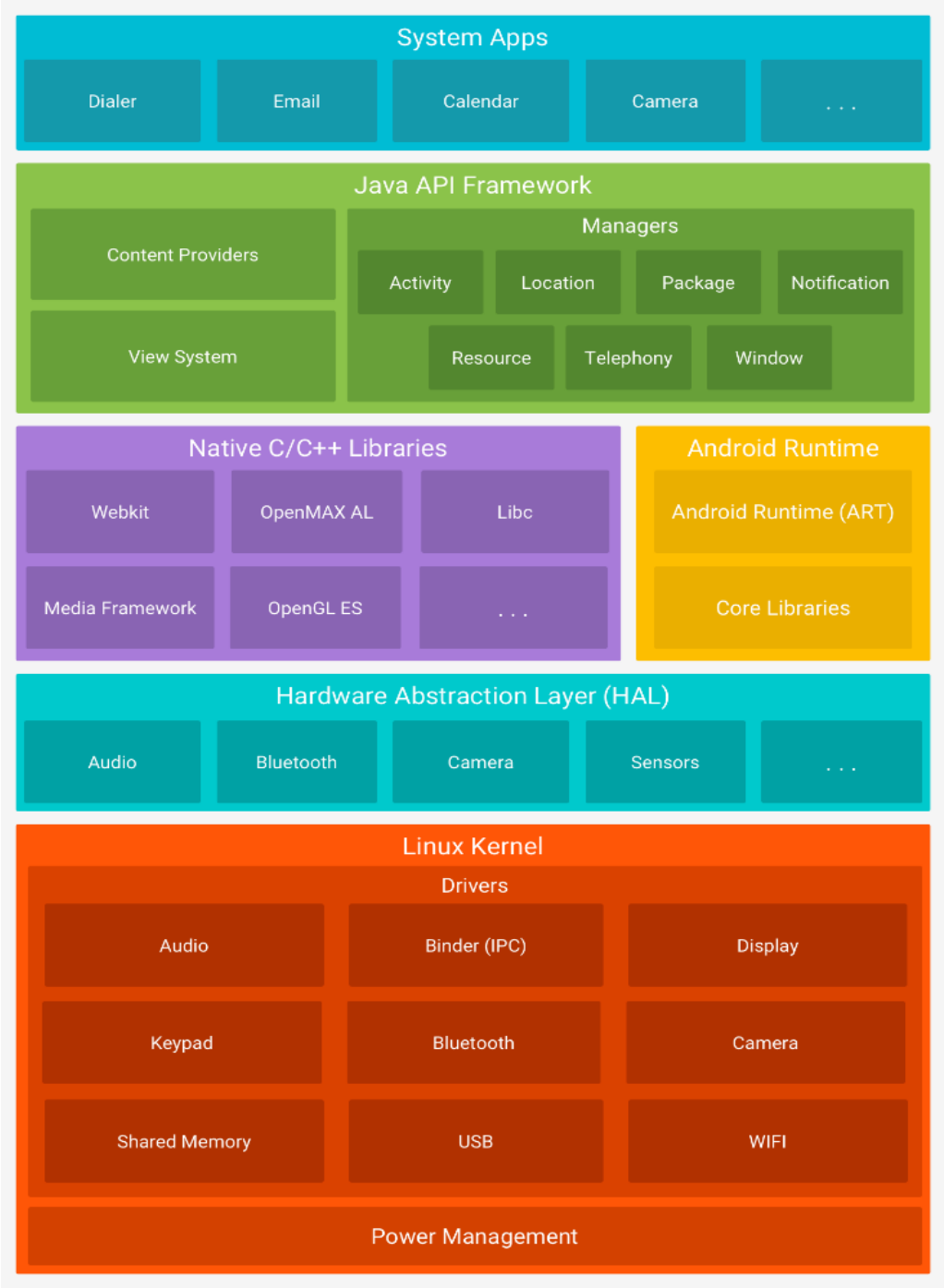


Рисунок 1.1 – Архітектура Android платформи

1) Шар System apps. Android постачається з набором основних програм для роботи з електронною поштою, SMS-повідомленнями, календарями, інтернетом, контактами тощо. Застосунки, що входять до складу платформи, не мають особливого статусу серед застосунків, які користувач обирає для встановлення. Таким чином, сторонній застосунок може стати веб-браузером за замовчуванням, SMS-месенджером або навіть клавіатурою за замовчуванням. Існують деякі винятки, наприклад, системний застосунок "Налаштування". Системні застосунки функціонують як застосунки для користувачів, так і надають ключові можливості, до яких розробники можуть отримати доступ зі своїх власних застосунків. Наприклад, якщо потрібно, щоб програма надсилала SMS-повідомлення, не потрібно створювати цю функцію самостійно. Замість цього можна викликати будь-яку вже встановлену програму для надсилання SMS-повідомлень, щоб доставити повідомлення вказаному одержувачу [2].

2) Шар Java Application Framework. Фреймворк застосунків надає кілька важливих класів, які використовуються для створення Android-застосунків. Він забезпечує загальну абстракцію для доступу до апаратного забезпечення і допомагає в управлінні користувацьким інтерфейсом з ресурсами програми. Загалом, він надає сервіси, за допомогою яких можемо створити певний клас і зробити його корисним для створення застосунків. Він включає в себе різні типи сервісів, такі як менеджер активності, менеджер сповіщень, система перегляду, менеджер пакетів і т.п., які є корисними для розробки нашого застосунку відповідно до передумов [3].

3) Рівень Application Framework надає багато високорівневих сервісів для застосунків у вигляді класів Java. Розробники застосунків можуть використовувати ці сервіси у своїх застосунках.

Фреймворк Android включає наступні ключові сервіси:

– Менеджер активності «Activity Manager»: Керує всіма аспектами життєвого циклу програми та стеком активностей.

– Постачальники контенту «Content Providers»: Дозволяє застосункам

публікувати та обмінюватися даними з іншими застосунками.

– Менеджер ресурсів «Resource Manager»: Надає доступ до некодованих вбудованих ресурсів, таких як рядки, налаштування кольорів і макети інтерфейсу користувача.

– Менеджер сповіщень «Notification Manager»: Дозволяє програмам показувати користувачеві попередження та сповіщення.

– Система перегляду «View System»: Розширюваний набір подань, що використовується для створення користувацьких інтерфейсів застосунків [3].

4) Шар Native C/C++ libraries. Багато основних системних компонентів і сервісів Android, таких як ART і HAL, побудовано на основі власного коду, який потребує власних бібліотек, написаних на C і C++. Платформа Android надає інтерфейси API фреймворку Java для розкриття функціональності деяких з цих власних бібліотек для застосунків. Для розробки Android потрібні нативні бібліотеки, і більшість з них мають відкритий вихідний код. Це набір базових бібліотек C/C++, а також бібліотек на основі Java, які підтримують розробку Android, таких як Graphics, Libc, SSL (Secure Socket Layer), SQLite, Media, Webkit, OpenGL (Open Graphic Library), Surface Manager тощо. Наведемо детальну інформацію про деякі ключові бібліотеки Android, які доступні для розробки під Android [3].

– Media бібліотека надає підтримку для відтворення та запису аудіо- та відеоформатів;

– Surface Manager бібліотека забезпечує функціональність керування дисплеєм;

– OpenGL (Open Graphic Library) та SGL (Scalable Graphics Library) бібліотеки використовуються для створення 2D та 3D графіки;

– SQLite бібліотека забезпечує підтримку баз даних;

– FreeType бібліотека забезпечує підтримку різноманітних шрифтів;

– SSL (Secure Socket Layer) бібліотека надає технології безпеки для встановлення зашифрованого з'єднання між веб-сервером і веб-браузером;

– WebKit надає набір класів, призначених для вбудовування можливостей веб-браузерів у застосунки .

4) Шар Android Runtime. Середовище виконання «Android Runtime» Android є однією з найважливіших частин Android. Воно містить такі компоненти, як основні бібліотеки та віртуальна машина Dalvik Virtual Machine (DVM). Головним чином, воно забезпечує основу для фреймворку застосунку та забезпечує роботу застосунку за допомогою основних бібліотек. Як і віртуальна машина Java Virtual Machine (JVM), віртуальна машина Dalvik Virtual Machine (DVM) - це віртуальна машина на основі реєстру, спеціально розроблена та оптимізована для Android, щоб гарантувати, що пристрій може ефективно запускати декілька екземплярів. Вона залежить від рівня ядра Linux для управління потоками та низькорівневого управління пам'яттю. Бібліотеки ядра дозволяють нам реалізовувати андроїд-застосунки за допомогою стандартних мов програмування JAVA або Kotlin [3].

5) Шар Hardware abstraction layer (HAL). Рівень абстракції апаратного забезпечення (HAL) надає стандартні інтерфейси, які розкривають апаратні можливості пристрою для високорівневого фреймворку Java API. HAL складається з декількох бібліотечних модулів, кожен з яких реалізує інтерфейс для певного типу апаратних компонентів, таких як камера або модуль Bluetooth. Коли API фреймворку робить виклик для доступу до апаратного забезпечення пристрою, система Android завантажує модуль бібліотеки для цього апаратного компонента [3].

6) Шар Linux Kernel. Ядро Linux - це серце архітектури Android. Воно керує всіма доступними драйверами, такими як дисплей, камера, Bluetooth, аудіо, пам'ять і т.д., необхідними під час виконання програми. Ядро Linux забезпечує рівень абстракції між апаратним забезпеченням пристрою та іншими компонентами архітектури Android. Воно відповідає за управління пам'яттю, живленням, пристроями тощо [3].

Особливостями ядра Linux:

– Безпека «Security»: ядро Linux відповідає за безпеку між застосунком і системою.

– Управління пам'яттю «Memory Management»: ефективно управляє пам'яттю, забезпечуючи свободу розробки наших застосунків.

– Управління процесами «Process Management»: добре керує процесом, розподіляє ресурси між процесами, коли вони їм потрібні.

– Мережевий стек «Network Stack»: уфективно управляє мережевим зв'язком.

– Модель драйверів «Driver Model»: гарантує, що програма працює належним чином на пристроях і апаратних засобах виробників, відповідальних за включення своїх драйверів у збірку Linux.

При створенні застосунків для Android потрібні різні компоненти пристрою Android, такі як камера, GPS тощо.

Для того, щоб використовувати ці можливості смартфона Android, повинні спочатку отримати дозвіл від користувача на використання чогось на його телефоні [3]. Крім того, дозволи мають різні рівні захисту (рис. 1.2), а саме:

1) Звичайні дозволи «Normal Permissions» - Значення за замовчуванням. Дозвіл із низьким рівнем ризику, який надає програмі-запитувачу доступ до ізольованих функцій на рівні програми з мінімальним ризиком для інших програм, системи або користувача. Система автоматично надає цей тип дозволу програмі-запитувачу під час встановлення, не запитуючи явного схвалення користувача, хоча користувач завжди має можливість переглянути ці дозволи перед встановленням [4].

2) Дозволи на підпис «Signature Permissions» - Дозвіл, який система надає тільки в тому випадку, якщо заявка, що запитує дозвіл, підписана тим самим сертифікатом, що і заявка, яка декларує дозвіл. Якщо сертифікати збігаються, система автоматично надає дозвіл, не повідомляючи користувача і не запитуючи його явного схвалення [4].

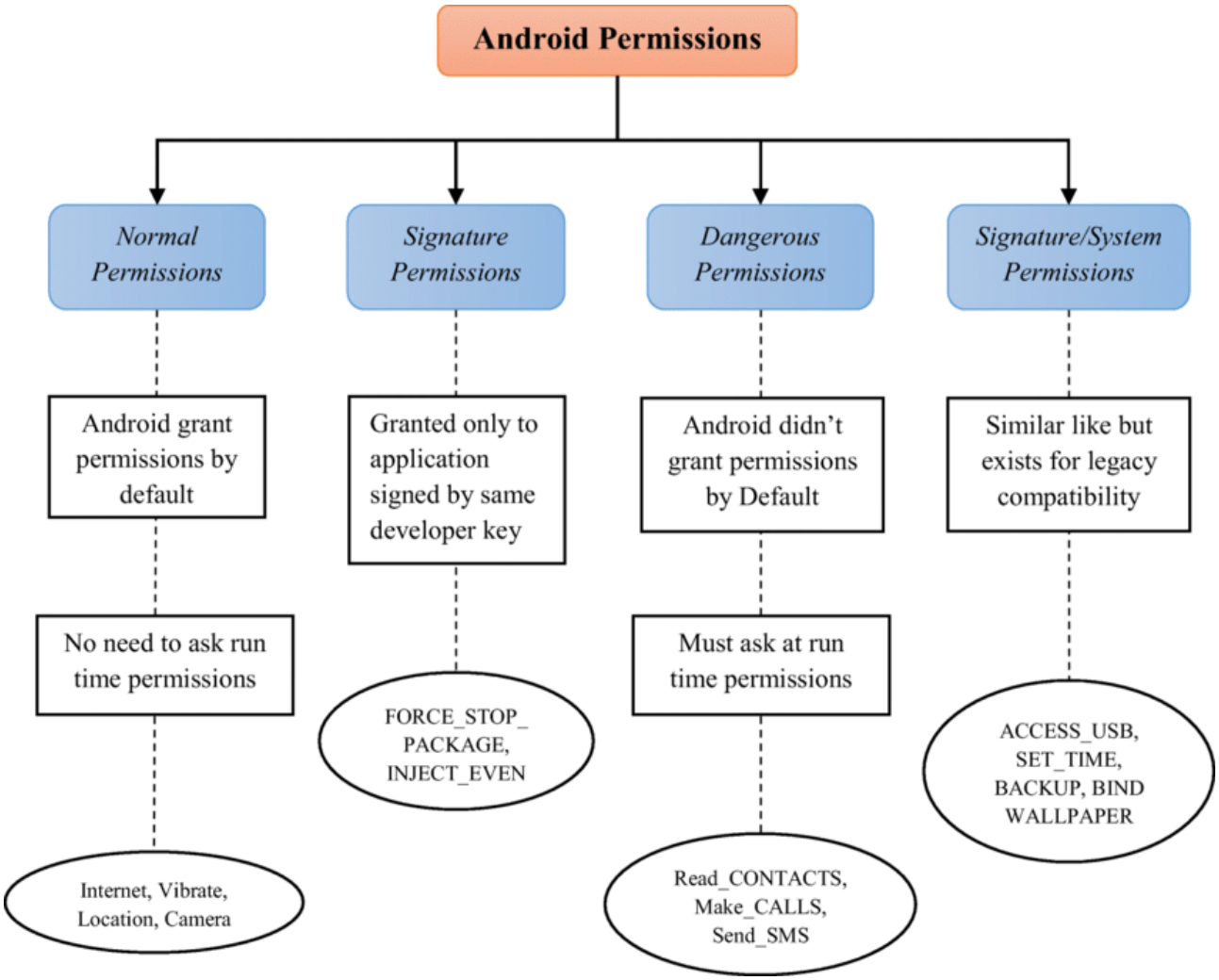


Рисунок 1.2 – Схема рівнів захисту для дозволів

3) Небезпечні дозволи «Dangerous Permissions» - Дозвіл з підвищеним ризиком, який надає програмі-запитувачу доступ до приватних даних користувача або контроль над пристроєм, що може негативно вплинути на користувача. Оскільки цей тип дозволів пов'язаний з потенційним ризиком, система може не надати його автоматично програмі, яка його запитує. Наприклад, будь-які небезпечні дозволи, запитувані програмою, можуть відображатися користувачеві і вимагати підтвердження перед продовженням роботи, або може бути застосований інший підхід, щоб уникнути автоматичного надання користувачем дозволу на використання таких засобів (рис 1.3) [4-5].

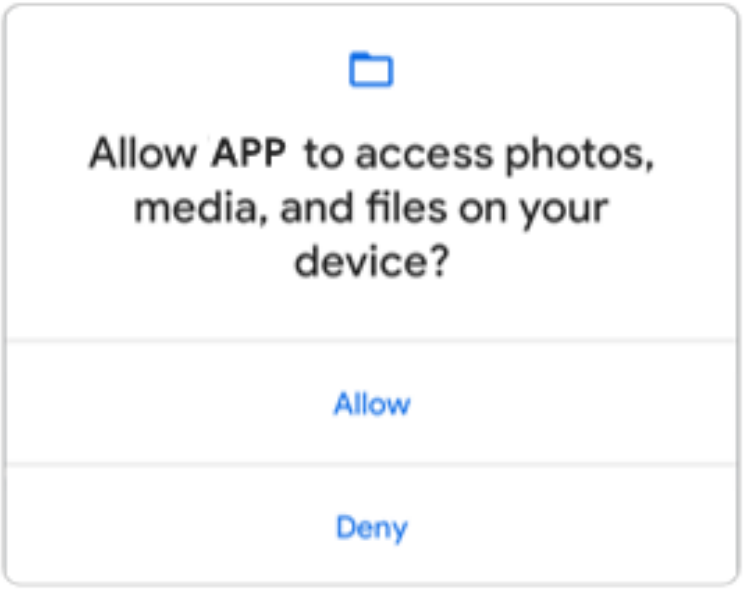


Рисунок 1.3 – Приклад запиту на отримання дозволу від системи, який з'являється, коли програма запитує дозвіл на виконання.

4) Рівень дозволу «Дозволи на підпис або Система» (signatureOrSystem). Стара назва "signatureOrSystem", який є застарілим з Android 6.0 Marshmallow. Дозвіл, який система надає лише програмам, що знаходяться у спеціальній папці в образі системи Android або підписані тим самим сертифікатом, що й програма, яка оголосила дозвіл. Дозвіл "signatureOrSystem" використовується у певних особливих ситуаціях, коли програми різних виробників вбудовано в образ системи, і їм потрібно надати спільний доступ до певних функцій, оскільки вони створюються спільно [6].

APK — це формат файлу, який використовується для розповсюдження та встановлення програм в операційній системі Android. Це стислий файл, який містить усі необхідні файли для роботи програми на пристрої Android, включаючи її код, ресурси та файл маніфесту Android. Файли APK схожі на інші формати файлів пакетів, наприклад файли .exe у Windows або файли .dmg у MacOS [7].

Користувачі можуть встановлювати програми на свої пристрої Android, завантажуючи їх з Інтернету, передаючи через USB або використовуючи магазин програм, наприклад Google Play Store. Після встановлення програму можна

запускати та використовувати як будь-яку іншу програму. Розробники можуть створювати APK за допомогою середовища розробки Android Studio, яке містить інструменти для створення, тестування та упаковки програм у APK. Розробники також можуть підписувати свої APK-файли цифровим підписом, щоб забезпечити автентичність і цілісність програми [7]

APK складається з кількох компонентів, необхідних для належної роботи програми на пристрої Android. Подробиці цих компонентів показано на рисунку 1.4.

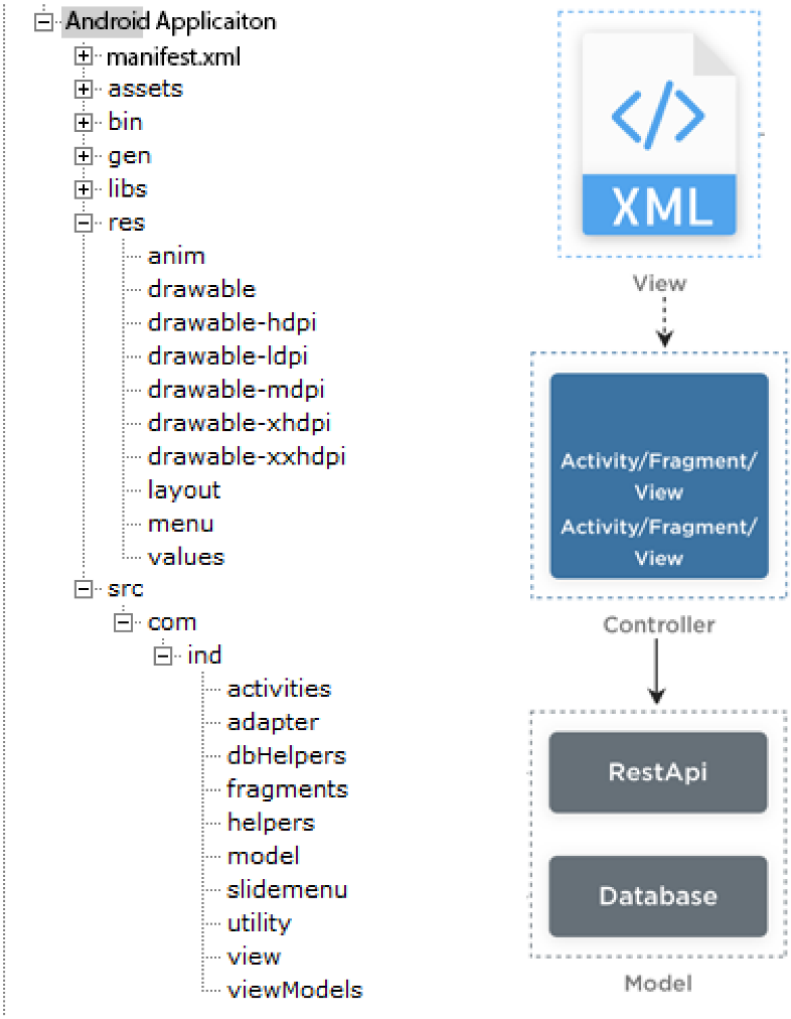


Рисунок 1.4 – Приклад файлової структури пакету Android-застосунків

– App code: Сюди входить вихідний код Java або Kotlin, який визначає функціональність програми. Цей код керує поведінкою програми та обробляє взаємодію з користувачем.

– Resources: Це файли, які використовує застосунок, наприклад,

зображення, макети та рядки. Ці ресурси визначають зовнішній вигляд програми та містять текст для різних мов.

– Android manifest: Це спеціальний XML-файл, який містить важливу інформацію про програму, наприклад, назву, версію та дозволи, які їй потрібні. Маніфест також визначає компоненти програми, такі як дії, сервіси та приймачі трансляцій.

– External libraries: Якщо програма використовує зовнішні бібліотеки, вони також пакуються в APK-файл. Ці бібліотеки надають додаткову функціональність і зазвичай є проектами з відкритим кодом, інтегрованими у програму.

– Assets: Це файли, які програма використовує, але не компілює, наприклад, шрифти, аудіо- та відеофайли. Вони зберігаються в APK в оригінальному форматі, і програма зчитує їх під час виконання.

– Android runtime (ART): Цей компонент запускає програму на пристроях Android; він перетворює байт-код програми в машинний код, щоб процесор пристрою міг її виконати.

AAB — це новий формат, запроваджений Google для розповсюдження програм Android. AAB — це нова заміна APK, представлена 6 серпня 2018 року з Android 9 [8], але вона розроблена, щоб бути більш ефективною та гнучкою. Основна відмінність між AAB і APK полягає в тому, що AAB містить увесь код і ресурси програми, але не містить скомпільований машинний код для всіх пристроїв. Натомість він містить код і ресурси програми у формі, яка може генерувати машинний код для певних пристроїв під час встановлення. Це робить програму меншою та ефективнішою, оскільки вона містить лише машинний код, необхідний для конкретного пристрою, на якому вона встановлюється. Крім того, AAB дозволяє розробникам створювати кілька файлів APK з одного проекту Android, які можуть бути націлені на різні пристрої, конфігурації екрана та мови, зменшуючи розмір кінцевого файлу APK і покращуючи взаємодію з користувачем [7].

Відповідно до бази даних загальних вразливостей і вразливостей (Common Vulnerabilities And Exposures), багато вразливостей було виявлено в операційній системі Android з моменту її створення. З 2009 року в ОС Android було виявлено близько 5000 вразливостей. На рисунку 1.5 наведено вразливості, виявлених за останні роки. Можна помітити, що кількість вразливостей, виявлених в ОС Android, неухильно зростала протягом багатьох років [7].

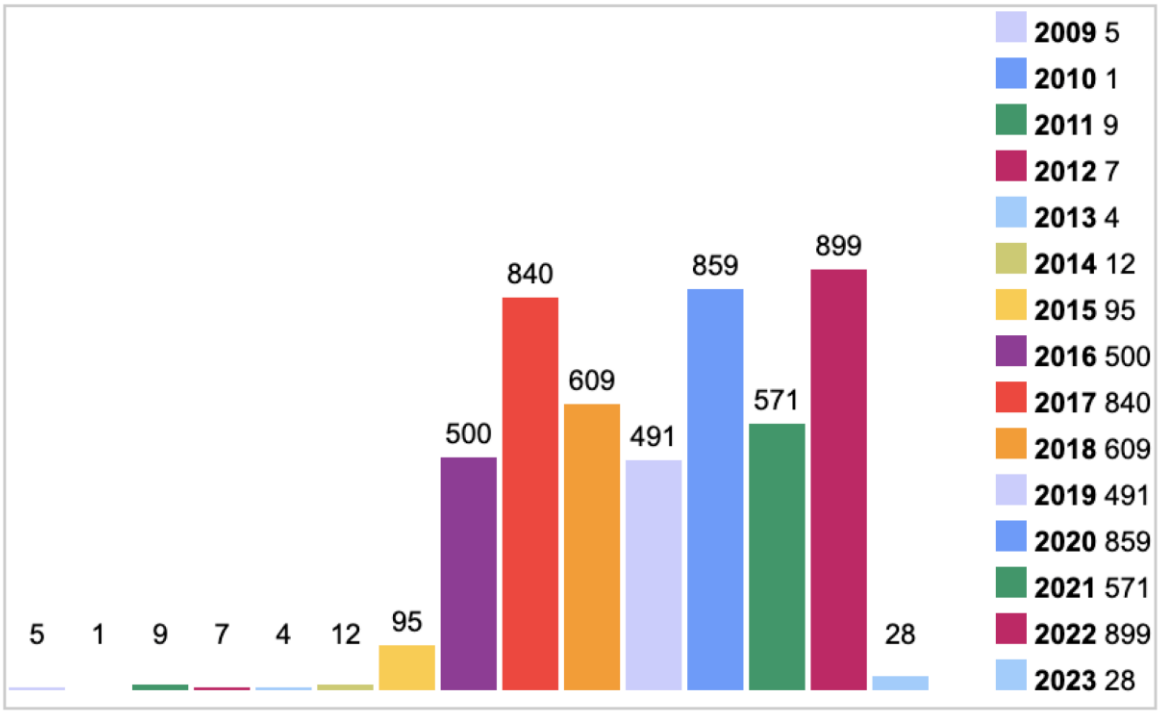


Рисунок 1.5 – Статистика вразливостей

Цифри демонструють постійні виклики, з якими стикається ОС Android щодо безпеки. Незважаючи на зусилля розробників і дослідників щодо підвищення безпеки операційної системи Android, кількість виявлених вразливостей продовжує зростати. Крім того, ці виявлені вразливості можна розділити на різні типи, такі як виконання коду, переповнення буфера та отримання інформації. Ця категоризація допомагає визначити, які типи вразливостей є найпопулярнішими та зустрічаються. На рисунку 1.6 наведено деталі вразливостей за різними типами. На рисунку показано, що найпоширенішими вразливими місцями, які призводять до експлуатації, є

виконання коду та переповнення буфера. Подробиці повідомлених типів вразливостей наведені нижче [7].

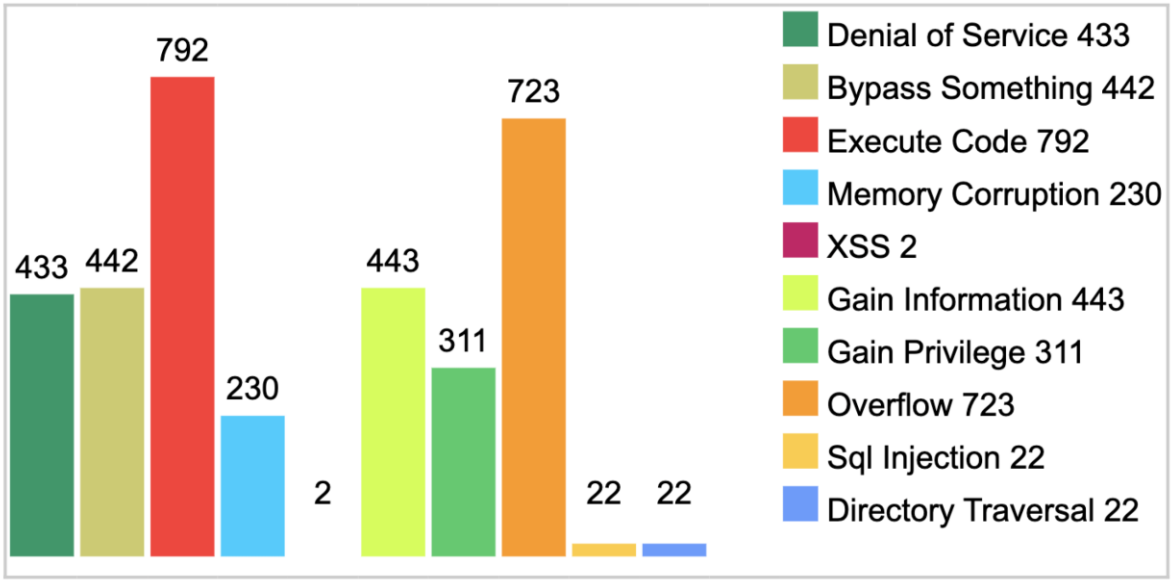


Рисунок 1.6 – Статистика вразливостей Android

Найпопулярніші вразливості ОС Android:

1. **Denial of service (DoS):** цей тип вразливості виникає, коли зловмисник переповнює систему трафіком або запитами, щоб перевантажити її ресурси і зробити її недоступною для легальних користувачів.

2. **Bypass something:** ця вразливість дозволяє зловмиснику обійти контроль безпеки, наприклад, аутентифікацію або авторизацію, отримати доступ до захищених ресурсів або виконати несанкціоновані дії.

3. **Execute code:** ця уразливість дозволяє зловмиснику виконати довільний код на цільовій системі або пристрої, що може призвести до крадіжки даних, несанкціонованого доступу або інших зловмисних дій.

4. **Memory corruption:** ця вразливість полягає у використанні помилок або недоліків в системі керування пам'яттю програми, таких як переповнення буфера або помилки використання після заборони, для отримання несанкціонованого доступу до даних або виконання шкідливого коду.

5. **Cross-site scripting (XSS):** ця вразливість дозволяє зловмиснику впроваджувати шкідливий код, наприклад Java чи Kotlin, на веб-сторінку чи

програму, що потенційно може призвести до крадіжки даних або інших шкідливих дій.

6. **Information disclosure:** ця вразливість дозволяє зловмисникам отримати доступ до конфіденційної інформації без авторизації, такої як паролі, особисті дані або інформація про конфігурацію системи.

7. **Privilege escalation:** ця вразливість дозволяє зловмиснику отримати вищий доступ або привілеї, ніж авторизовано, потенційно дозволяючи йому виконувати зловмисні дії або отримувати доступ до конфіденційних даних.

8. **Buffer overflow:** ця вразливість виникає, коли зловмисник вводить більше даних у буфер пам’яті програми, ніж він може обробити, що потенційно може призвести до виконання довільного коду або збою системи.

9. **SQL injection:** ця вразливість дозволяє зловмиснику впроваджувати шкідливий код SQL у веб-програму або базу даних, що потенційно може призвести до несанкціонованого доступу або модифікації даних.

10. **Directory traversal:** ця вразливість передбачає використання помилки перевірки шляху до файлу веб-програми для отримання несанкціонованого доступу до файлів або каталогів за межами передбаченої програми.

Цю інформацію можна використовувати для визначення пріоритетів у усуненні вразливостей і виявлення закономірностей у типах вразливостей, які найчастіше націлені. Крім того, він може надавати інформацію про те, як розвивається галузь, і дозволяє вживати профілактичних заходів для усунення вразливостей, які, швидше за все, стануть ціллю в майбутньому [7].

Google Play Store – це офіційний магазин програм для пристроїв Android, де користувачі можуть завантажувати та встановлювати програми, ігри та інші види програмного забезпечення [9]. Програми в магазині розробляються сторонніми розробниками та перевіряються Google перед тим, як бути загальнодоступними. Google запровадив інструмент Play Protect для сканування програм на наявність зловмисного програмного забезпечення та іншого шкідливого вмісту перед публікацією в магазині. Це включає статичний і

динамічний аналіз застосунків і перевірку будь-якої відомої шкідливої поведінки. Однак, незважаючи на ці заходи, зловмисники проникають у магазин. Існує кілька підходів, за допомогою яких зловмисники можуть завантажувати та розповсюджувати зловмисне програмне забезпечення через Google Play Store, деталі яких наведено нижче. [7].

1. **Динамічне завантаження коду:** зловмисники можуть використовувати методи динамічного завантаження коду, такі як відображення Java або Android DexClassLoader, для завантаження та виконання коду під час виконання. Це можна використовувати для завантаження та виконання додаткового коду або зловмисних корисних навантажень після встановлення програми, які можуть бути приховані в законному коді програми або завантажені з віддаленого сервера. Це може дозволити зловмисникам уникнути виявлення сканерами безпеки, оскільки шкідливий код може бути відсутнім у початковій версії програми.

2. **Атака інкрементних шкідливих оновлень (IMUTA):** зловмисники можуть використовувати інкрементні оновлення для поступового додавання шкідливого коду до програми з часом. Це можна зробити за допомогою програми дроппера, яка спочатку здається законною, але пізніше завантажує та встановлює додаткові компоненти або корисні навантаження. Зловмисники також можуть використовувати методи обфускації коду та динамічного завантаження коду, щоб додати шкідливий код у спосіб, який важко виявити, наприклад оновлення. Це може дозволити зловмисникам уникнути виявлення сканерами безпеки та продовжити термін служби шкідливого програмного забезпечення.

3. **Обфускація коду:** обфускація полягає у зміні вихідного коду програми, щоб зробити його важчим для розуміння чи аналізу. Зловмисники можуть використовувати методи обфускації, такі як перейменування змінних і класів, додавання небажаного коду або використання шифрування, щоб приховати функціональність і призначення зловмисного коду. Це може ускладнити дослідникам безпеки виявлення та аналіз шкідливих програм.

4. **Перепакування:** зловмисники можуть використовувати методи перепакування, щоб взяти законну програму та додати шкідливий код. Це можна зробити шляхом декомпіляції програми, додавання зловмисного коду, а потім перекомпіляції та повторної реєстрації програми. Потім переупаковану програму можна завантажити в магазин Google Play, і користувачам можуть маніпулювати її завантаження, оскільки вона може виглядати законною та мати хороші відгуки.

5. **Соціальна інженерія:** соціальна інженерія змушує людей виконувати певні дії або розголошувати конфіденційну інформацію. Деякі кіберзлочинці використовують соціальну інженерію, щоб обманом спонукати користувачів завантажувати та встановлювати програми, які здаються легітимними, але є шкідливими програмами. Ці програми можуть виглядати як популярні програми чи ігри, але містити зловмисне програмне забезпечення, яке може викрасти особисту інформацію або виконати інші шкідливі дії.

У магазині Google Play розміщено численні сімейства зловмисного програмного забезпечення, яке було ідентифіковано, повідомлено та опубліковано. Google виявив і заблокував багато сімейств зловмисного програмного забезпечення для Android, але багато останніх інцидентів все ще свідчать про лазівки та вразливості в механізмах безпеки, реалізованих Google Play Store [7,9].

У таблиці 1.1 наведено 10 сімейств шкідливих програм, які поширюються за допомогою Google Play Store. Ці сімейства зловмисних програм не обмежуються однією програмою чи обліковим записом, а їх код може бути пов’язаний із сотнями програм, які користувачі можуть мимоволі завантажити. Таким чином, надзвичайно важливо, щоб користувачі та розробники виявляли обережність і старанність під час використання та створення програм, відповідно, і щоб магазин Google Play продовжував впроваджувати надійні заходи безпеки для виявлення та запобігання поширенню зловмисного програмного забезпечення [7,9].

Таблиця 1.1 - Список найпоширеніших сімейств шкідливих програм для Android, виявлених у Google Play Store.

Назва	Кількість застосунків	Загальний відсоток (%)
Airpush	1574	23.8
Plankton	722	10.9
Adwo	583	9.0
Kuguo	492	7.4
Leadbolt	298	4.5
Dowgin	261	3.9
Fakeinst	214	3.2
Gingermaster	194	2.9
Wapsx	192	2.8
Youmi	110	1.7

Ці сімейства зловмисного програмного забезпечення можуть перебувати в різних програмах Google Play Store кількома способами. Щойно зловмисне програмне забезпечення завантажується та встановлюється на пристрій користувача, воно може виконувати різні шкідливі дії. Це може включати викрадення особистої інформації, показ небажаної реклами або навіть контроль над пристроєм. Ці атаки зловмисного програмного забезпечення може бути важче виявити за допомогою Google Play Protect та інших мобільних антивірусів, оскільки вони вбудовані в програми та можуть не мати жодних видимих ознак зловмисної поведінки. Зловмисне програмне забезпечення також може запускатися лише через певний час, що ускладнює його виявлення антивірусам [7,9].

Зловмисне програмне забезпечення можна виявити, проаналізувавши дозволи, які вони запитують під час встановлення на пристрої Android. Аналіз дозволів, які запитує певна програма, дає змогу визначити, чи є вона потенційно зловмисною. Наприклад, якщо програма запитує дозволи, не пов’язані з її

призначеними функціями, як-от можливість доступу до контактів користувача чи журналів викликів, це може означати, що програма шкідлива. Крім того, якщо програма запитує дозволи, які надають їй доступ до конфіденційної інформації, як-от даних про місцезнаходження, це може бути ознакою шкідливого програмного забезпечення. Таблиця 1.2 надає ресурс для виявлення зловмисного програмного забезпечення, перераховуючи дозволи, які найчастіше запитує зловмисне програмне забезпечення після встановлення. Це може бути корисним ресурсом для виявлення зловмисного програмного забезпечення, оскільки містить перелік дозволів, які найчастіше запитує зловмисне програмне забезпечення після встановлення [7,9].

Таблиця 1.2 - Список найпоширеніших дозволів, запитуваних шкідливим програмним забезпеченням Android.

Ім’я дозволу	Відсоток використання	Група дозволів	Класифікація на основі дозволів
INTERNET	98.8	Мережа	Небезпечно
ACCESS_NETWORK_STATE	94.4	Мережа	Нормально
READ_PHONE_STATE	82.7	Телефонний дзвінок	Небезпечно
WRITE_EXTERNAL_STORAGE	70.4	Сховище	Небезпечно
ACCESS_COARSE_LOCATION	56.2	Розташування	Небезпечно
ACCESS_WIFI_STATE	54.9	Мережа	Нормально
ACCESS_FINE_LOCATION	53.7	Розташування	Небезпечне
Ім’я дозволу	Відсоток використання	Група дозволів	Класифікація на основі дозволів
GET_ACCOUNTS	29.0	Облікові записи	Нормально
C2D_MESSAGE	24.7	-	Підпис
RECEIVE_BOOT_COMPLETED	24.7	Інформація про програму	Нормально
SYSTEM_ALERT_WINDOW	17.3	Відображення	Небезпечно
CALL_PHONE	14.2	Дзвінки на телефон	Небезпечно

Продовження табл. 1.2 - Список найпоширеніших дозволів, запитуваних шкідливим програмним забезпеченням Android.

Ім’я дозволу	Відсоток використання	Група дозволів	Класифікація на основі дозволів
CAMERA	13.0	Камера	Небезпечно
RECORD_AUDIO	12.3	Мікрофон	Небезпечно
EAD_HISTORY_BOOKMARKS	11.7	Закладки	Небезпечно
SEND_SMS	11.7	Повідомлення	Небезпечно
READ_EXTERNAL_STORAGE	14.8	Зберігання	Нормально

Проаналізувавши таблиці 1.1 і 1.2 , експерти з безпеки та дослідники можуть ідентифікувати потенційно шкідливі програми, аналізуючи дозволи, які вони запитують. Також важливо зазначити, що не всі програми, які запитують ці дозволи, є зловмисними, але це може бути червоним прапорцем, який потребує подальшого дослідження.

1.2 Аналіз методів та засобів тестування на проникнення мобільних застосунків

Тестування на проникнення мобільних застосунків охоплює цілий ряд підходів, кожен з яких адаптований до конкретних вимог тестування. Ці методології дають чітке уявлення про стан безпеки застосунку та його вразливості. Існує три основні типи тестування на проникнення мобільних застосунків:

– Тестування «чорної скриньки» проводиться без наявності у тестувальника будь-якої інформації про застосунок, що тестується. Цей процес іноді називають «тестуванням з нульовим знанням». Основна мета цього тесту полягає в тому, щоб дозволити тестувальнику поводитися як справжній зловмисник у сенсі вивчення можливих способів використання загальнодоступної та відкритої інформації.

– Тестування за принципом «білої скриньки» (іноді його називають «перевіркою повного знання») є повною протилежністю тестуванню за «чорною

скринькою» в тому сенсі, що тестувальник має повні знання про програму. Знання можуть охоплювати вихідний код, документацію та діаграми. Цей підхід дозволяє тестувати набагато швидше, ніж тестування за допомогою чорної скриньки, завдяки своїй прозорості, а з додатковими знаннями, отриманими тестувальником, можна створювати набагато складніші та детальніші тестові випадки.

– Тестування «сірої скриньки» – це все тестування, яке знаходиться між двома вищезгаданими типами тестування: деяка інформація надається тестувальнику (зазвичай лише облікові дані), а інша інформація призначена для виявлення. Цей тип тестування є цікавим компромісом щодо кількості тестів, вартості, швидкості та обсягу тестування. Тестування сірого ящика є найпоширенішим видом тестування в індустрії безпеки [10].

Тестування на проникнення мобільних застосунків - це процес виявлення та вирішення потенційних зразків у безпеці застосунків для мобільних пристроїв. Цей процес може бути складним і включати в себе кілька етапів. До загальних етапів тестування на проникнення мобільних застосунків відбувається в 6 етапів (рис. 1.7):

1) підготовка;

2) збір розвідувальної інформації;

3) складання мапи застосунку;

4) експлуатація;

5) звітування.

Під час етапу підготовки вирішуються такі задачі:

– визначається обсяг тестування безпеки, включаючи визначення відповідних засобів контролю безпеки, цілей тестування організації та конфіденційних даних. Загалом, підготовка включає всю синхронізацію з клієнтом, а також юридичний захист тестувальника (який часто є третьою стороною).

На наступному етапі виконується збір розвідувальної інформації:

– виконується аналіз середовища та архітектурного контексту програми для отримання загального розуміння контексту;

Етап складання мапи застосунку – базується на інформації з попередніх етапів; може доповнюватися автоматизованим скануванням і ручним дослідженням застосунку. Картування забезпечує глибоке розуміння застосунку, його точок входу, даних, які він зберігає, та основних потенційних вразливостей. Потім ці вразливості можна ранжувати відповідно до шкоди, яку може спричинити їхня експлуатація, щоб тестувальник безпеки міг визначити їхній пріоритет. Ця фаза включає створення тестових кейсів, які можуть бути використані під час виконання тесту.

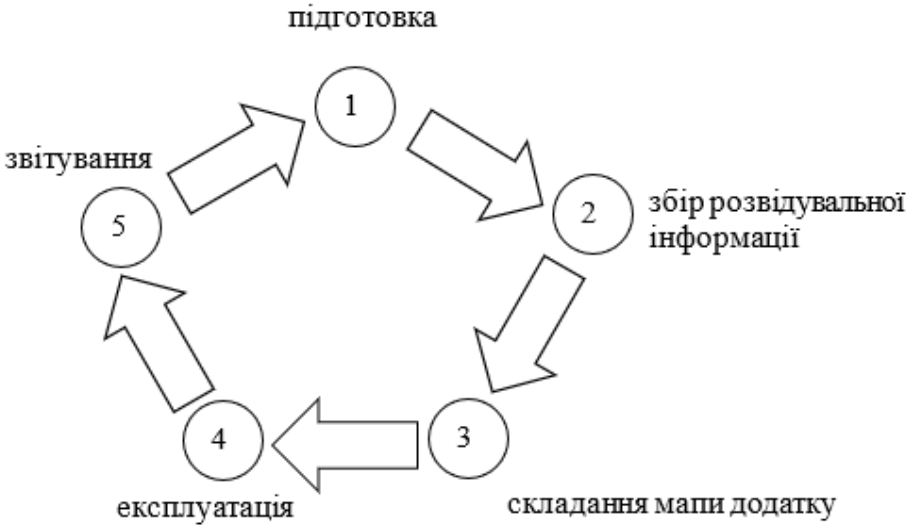


Рисунок 1.7 – Схема процесу тестування на проникнення мобільних застосунків

Після сканування мапи застосунку відбувається етап експлуатації:

– на цьому етапі тестувальник безпеки намагається проникнути в застосунок, використовуючи вразливості, виявлені на попередньому етапі. Цей етап необхідний для визначення того, чи є уразливості реальними та справді позитивними.

Фінальний етап звітування є важливим для клієнта, тестувальник безпеки повідомляє про вразливості. Це включає в себе детальний процес експлуатації, класифікує тип уразливості, документує ризик, якщо зловмисник зможе скомпрометувати ціль, і окреслює, до яких даних тестер мав незаконний доступ [11].

Тестування на проникнення мобільних застосунків вимагає відповідності до різних стандартів та керівництв. Деякі з найбільш важливих стандартів та документів, які можуть використовуватися при проведенні тестування на проникнення мобільних застосунків:

– Open Web Application Security Project (OWASP) - Mobile Security Testing Guide (MSTG): MSTG надає детальний і практичний посібник про те, як проводити тестування безпеки мобільних застосунків, що охоплює платформи Android і iOS. MSTG охоплює такі теми, як архітектура мобільних застосунків, зберігання даних, криптографія, автентифікація, мережевий зв'язок, зворотне проектування, аналіз коду та тестування на проникнення. MSTG також надає контрольний список вимог безпеки та найкращих практик, які можуть бути використані для оцінки та покращення безпеки мобільних застосунків [12].

– NIST SP 800-163: Vetting the Security of Mobile Applications: Стандарт надає докладні вказівки щодо оцінки безпеки мобільних застосунків, включаючи тести на проникнення [13].

– Mobile Application Security Testing (MAST) Framework: Це ресурс, розроблений міжнародною організацією банківського сектору, який надає загальний підхід до тестування безпеки мобільних застосунків [14].

– NIST SP 800-163 Revision 1 «Vetting the Security of Mobile Applications» є важливим оновленням керівництва NIST з перевірки та безпеки мобільних застосунків. В оригінальному документі (січень 2015 року) детально описані процеси, за допомогою яких організації оцінюють мобільні додатки на предмет вразливостей кібербезпеки. Revision 1 розширює початковий документ, досліджуючи ресурси, які можуть бути використані для формування вимог організації до безпеки мобільних застосунків. Сюди входять огляди відповідної документації Національного партнерства із захисту інформації (NIAP), Проекту безпеки відкритих веб-застосунків (OWASP), корпорації MITRE і Національного інституту стандартів і технологій (NIST) [15-16].

– MITRE ATT&CK: Це глобальна доступна база знань про тактику і

методи противника, заснована на реальних спостереженнях. База знань ATT&CK використовується як основа для розробки конкретних моделей і методологій протидії загрозам у приватному секторі, в уряді, а також у спільноті розробників продуктів і послуг у сфері кібербезпеки.

В таблиці 1.3 наведено методи тестування на проникнення мобільних застосунків, їх явні переваги та недоліки.

Таблиця 1.3 – Методи тестування мобільних застосунків на проникнення.

Назва метода	OWASP - MSTG	NIST SP 800-163 Revision 1 «Vetting the Security of Mobile Applications»	Mobile Application Security Testing (MAST)	MITRE ATT&CK
Наявність документації	+	+	+	+
Наявність чіткої документації	+	- (має посилання на OWASP та MITRE ATT&CK)	- (має посилання на OWASP та MITRE ATT&CK)	+
Назва метода	OWASP - MSTG	NIST SP 800-163 Revision 1 «Vetting the Security of Mobile Applications»	Mobile Application Security Testing (MAST)	MITRE ATT&CK
Легкість сприймання документації	+ (зрозуміло що потрібно робити з інформацією)	- (не зрозуміло що потрібно робити з інформацією)	-	- (надмірна кількість посилань на різні ресурси)
Наявність засобів тестування на проникнення	+	-	-	+

З наведеної таблиці можна визначити, що в методах відсутня інформація того як потрібно розробляти звіт з тестування. Для розробки власного методу тестування за основу взято MSTG, MSTG є найкращим та най зрозумілішим методом тестування мобільного застосунку на проникнення через наявність чіткої документації, наявність засобів тестування на проникнення та легкість сприймання документації.

Стандарт перевірки безпеки мобільних застосунків «Mobile Application Security Verification Standard» (MASVS) – це комплексний стандарт безпеки,

розроблений OWASP. Стандарт розділений на різні групи, які представляють найбільш критичні області поверхні мобільних атак. Ці контрольні групи, позначені MASVS-XXXXX, надають загальні рекомендації та стандарти для наступних областей:

1. MASVS-STORAGE: Безпечне зберігання конфіденційних даних на пристрої (data-at-rest);

1.1 MASVS-STORAGE-1 Застосунок надійно зберігає конфіденційні дані;

1.2 MASVS-STORAGE-2 Застосунок запобігає витоку конфіденційних даних.

2. MASVS-CRYPTO: Криптографічні функції, що використовуються для захисту конфіденційних даних;

2.1 MASVS-CRYPTO-1 Застосунок використовує поточну надійну криптографію та використовує її відповідно до галузеві передові практики;

2.2 MASVS-CRYPTO-2 Застосунок керує ключами відповідно до передових галузевих практик.

3. MASVS-AUTH: Механізми автентифікації та авторизації, що використовуються мобільним додатком;

3.1 MASVS-AUTH-1 Застосунок використовує безпечні протоколи автенти-фікації та авторизації та слідує відповідні найкращі практики;

3.2 MASVS-AUTH-2 Застосунок безпечно виконує локальну автентифікацію відповідно до найкращих практик платформи.;

3.3 MASVS-AUTH-3 Застосунок захищає конфіденційні операції за допомогою додаткової автентифікації.

4. MASVS-NETWORK: Безпечний мережевий зв'язок між мобільним додатком і віддаленими кінцевими точками (дані в дорозі);

4.1 MASVS-NETWORK-1 Застосунок захищає весь мережевий трафік відповідно до поточних передових практик;

4.2 MASVS-NETWORK-2 Застосунок виконує закріплення ідентифікаційних даних для всіх віддалених кінцевих точок під контролем розробника.

5. MASVS-PLATFORM: Безпечна взаємодія з базовою мобільною платформою та іншими встановленими додатками;

5.1 MASVS-PLATFORM-1 Застосунок безпечно використовує механізми IPC;

5.2 MASVS-PLATFORM-2 Застосунок безпечно використовує WebViews.

5.3 MASVS-PLATFORM-3 Застосунок безпечно використовує інтерфейс користувача.

6. MASVS-CODE: Найкращі практики безпеки для обробки даних та підтримання актуальності програми;

6.1 MASVS-CODE-1 Застосунку потрібна найновіша версія платформи;

6.2 MASVS-CODE-2 Застосунок має механізм примусового оновлення програм;

6.3 MASVS-CODE-3 Застосунок використовує лише програмні компоненти без відомих вразливостей;

6.4 MASVS-CODE-4 Застосунок перевіряє та дезінфікує всі ненадійні дані;

7. MASVS-RESILIENCE: Стійкість до спроб зворотного інжинірингу та втручання;

7.1 MASVS-RESILIENCE-1 Застосунок перевіряє цілісність платформи.

7.2 MASVS-RESILIENCE-2 Застосунок реалізує механізми захисту від втручання;

7.3 MASVS-RESILIENCE-3 Застосунок реалізує механізми антистатичного аналізу;

7.4 MASVS-RESILIENCE-4 Застосунок реалізує методи антидинамічного аналізу.

Тестування на проникнення мобільних застосунків вимагає інструментів для виконання поставленої задачі. До найпопулярніших інструментів тестування мобільних застосунків відносяться:

1) OWASP ZAP (Zed Attack Proxy) - це універсальний інструмент для тестування безпеки веб та мобільних застосунків [12].

- може виконувати автоматизоване сканування для виявлення поширених вразливостей, таких як SQL-ін'єкції, міжсайтовий скриптинг (XSS) тощо;
- пропонує функції перехоплення і модифікації запитів і відповідей для тестування безпеки застосунків;
- надає зручний інтерфейс як для початківців, так і для досвідчених тестувальників.

2) MobSF (Mobile Security Framework) - це комплексний інструмент для оцінки безпеки мобільних застосунків [39].

- підтримує статичний аналіз шляхом декомпіляції та аналізу файлів APK (Android) та IPA (iOS) ;
- виконує динамічний аналіз, взаємодіючи з мобільним застосунком і відстежуючи його поведінку;
- виявляє вразливості, такі як незахищене зберігання даних, незахищене мережеве з'єднання тощо.

3) Drozer спеціально розроблений для тестування безпеки Android [28].

- може виявляти вразливості, такі як неправильні дозволи, експортовані компоненти та небезпечне зберігання файлів;
- дозволяє тестувальникам взаємодіяти з пристроями та програмами Android через інтерфейс командного рядка;
- підтримує різні модулі для тестування конкретних компонентів застосунків Android.

4) Frida - це динамічний інструментарій для Android та iOS [29].

- дозволяє аналізувати та маніпулювати поведінкою застосунків у реальному часі, включаючи перехоплення функцій та написання сценаріїв.

- корисний для аналізу та модифікації поведінки запущених мобільних застосунків під час виконання.

5) AppUse - це віртуальна машина, яка постачається з набором попередньо встановлених інструментів для тестування безпеки мобільних застосунків [40];

- забезпечує контрольоване середовище для тестування застосунків Android та їхніх функцій безпеки.

6) QARK (Quick Android Review Kit) - зосереджується на оцінці безпеки застосунків для Android [41].

- може виявити такі вразливості, як незахищене зберігання даних, виконання коду та неналежні дозволи;

- створює детальні звіти з інформацією про знайдені вразливості.

7) AndroBugs Framework [42].

- сканує застосунки для android на наявність проблем з безпекою;

- виявляє вразливості, такі як виконання коду, незахищені компоненти та витік даних.

- пропонує зручний інтерфейс для сканування та створення звітів.

8) Xposed Framework - дозволяє тестувальникам створювати та встановлювати модулі, які можуть перехоплювати та змінювати поведінку застосунків Android. [43].

- корисно для налаштування і тестування поведінки застосунків і функцій безпеки.

9) Burp Suite Mobile Assistant - це розширення Burp Suite використовується для тестування безпеки API-інтерфейсів мобільних застосунків. [44].

- перехоплює та аналізує запити і відповіді, щоб виявити уразливості безпеки у викликах API, що здійснюються мобільними застосунками.

В таблиці 1.4 наведено інструменти тестування на проникнення для мобільних застосунків, їх основні функції, основні атаки, на яких вони спеціалізуються, уразливості, на які вони націлені, вартість і тестові платформи.

Таблиця 1.4 – Засоби тестування мобільних застосунків на проникнення

Назва засобу	Спеціалізація інструменту тестування	Вартість	Платформи тестування
OWASP ZAP	Інструмент для виявлення та експлуатації вразливостей веб-застосунків, таких як SQL-ін'єкції, XSS, CSRF та брутфорс.	Безкоштовний	Android, iOS
MobSF	Інструмент для виявлення: - Статичний аналіз коду; - Динамічний аналіз коду; - Інтерактивний аналіз коду; - Аналіз бінарних файлів та метаданих.	Безкоштовний	Android, iOS, Web
Drozer	Динамічний аналізу дозволяє взаємодіяти з компонентами застосунку, виконувати команди на пристрої, виявляти вразливості, такі як небезпечні наміри, небезпечні URI, небезпечні збереження даних.	Безкоштовний	Android
Frida	Дозволяє вбудовувати свій власний код у процеси застосунків, змінювати їх поведінку, виявляти вразливості, такі як небезпечні виклики API, витоки даних, шифрування.	Безкоштовний	Android, iOS
AppUse	Віртуальна машина, яка містить набір інструментів для тестування безпеки Android-застосунків, таких як Drozer, Burp Suite, apktool, dex2jar, JD-GUI, Nmap, Metasploit.	Безкоштовний	Android
QARK	Інструмент для статичного аналізу безпеки Android-застосунків, який виявляє вразливості, такі як небезпечні наміри, небезпечні збереження даних, небезпечні виклики API, небезпечні використання криптографії.	Безкоштовний	Android
Xposed Framework	Інструмент для модифікації поведінки застосунків без зміни їх коду, який дозволяє використовувати різні модулі для зміни параметрів, функцій, інтерфейсу.	Безкоштовний	Android
MobXSS	Інструмент для виявлення та експлуатації вразливостей XSS в мобільних застосунках, які використовують WebView, за допомогою вбудованого сервера, який генерує зловмисні URL-адреси.	Безкоштовний	Android, iOS, Web

Продовження табл. 1.4 – Засоби тестування мобільних застосунків на проникнення.

Назва засобу	Спеціалізація інструменту тестування	Вартість	Платформи тестування
Burp Suite Mobile Assistant	Інструмент, який дозволяє інструментувати мобільні застосунки для використання Burp Suite, який є інтегрованою платформою для тестування безпеки веб-застосунків.	Платний - Burp Suite Professional - 449$ рік.	Burp Suite Mobile Assistant
AndroBugs Framework	Інструмент для статичного аналізу безпеки Android-застосунків, який виявляє вразливості, такі як небезпечні виклики API, небезпечні збереження даних, небезпечні використання криптографії, небезпечні використання SSL, небезпечні використання WebView.	Безкоштовний	Android

Ці інструменти охоплюють різні аспекти тестування безпеки мобільних застосунків, від статичного і динамічного аналізу до маніпуляцій під час виконання і тестування API. Залежно від конкретних потреб і платформ, які ви тестуєте, ви можете вибрати найбільш підходящі інструменти для тестування мобільних застосунків на проникнення.

1.3 Формалізація вимог та постановка задачі

Одним із механізмів захисту мобільних застосунків є тестування на проникнення. Було проаналізовано методи та засоби тестування на проникнення мобільних застосунків. Створення власної інформаційної технології тестування на проникнення дозволяє зекономити час на пошук методів тестування, засобів тестування та оптимізувати даний процес, адже ці стандарти зазвичай досить об’ємні. В залежності від типу складності та набору функцій застосунків методи та засоби тестування відрізняються.

В ході аналізу булу виявлено що OWASP є найкращою збіркою методологій тестуванням мобільного застосунку на проникнення, але мінусом кожного із методів, є, те що, вони не надають інформацію як

створювати звіт з тестування мобільного застосунку та який метод та засіб потрібно використовувати для різних типів застосунків.

У даній роботі об’єктом виступає процес вдосконалення методів тестування на проникнення мобільних застосунків під час здійснення тестування на проникнення.

Головною метою є покращення процесу тестування на проникнення на етапах збору даних та експлуатації застосунку за допомогою засобів тестування.

Після аналізу засобів для тестування MASVS, було сформовано набір вимого до технології тестування мобільного застосунку на проникнення:

- тестування конфігурації криптографічних стандартних алгоритмів;
- перевірка локального сховища на конфіденційні дані;
- перевірка конфіденційних даних;
- тестування резервних копій на конфіденційні дані;
- тестування шифрування даних у мережі;
- тестування дозволів програми;
- тестування локального сховища для перевірки введених даних.

На основі проаналізованих даних, визначено наступні задачі:

– Аналіз системи безпеки: Провести докладний аналіз системи безпеки мобільного застосунку для виявлення потенційних вразливостей.

– Тестування на проникнення: Розробити та здійснити тестування на проникнення з використанням різноманітних методів атак для перевірки ефективності системи захисту.

– Виявлення вразливостей та їх документування: Виявити та документувати уразливості, включаючи опис можливих сценаріїв атак та рекомендації щодо їх виправлення.

– Розробка рекомендацій: Розробити рекомендації для вдосконалення системи безпеки мобільного застосунку з урахуванням результатів тестування.

2 РОЗРОБКА ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ

2.1 Структура інформаційної технології

Інформаційна технологія – це комплекс методів і засобів, завдяки яким реалізується певна задача. Інформаційна технологія складається з процесів, що являють собою незалежні частини. Такий підхід дозволяє виявляти проблеми на ранніх стадіях розробки та швидко їх вирішувати.

Інформаційна технологія тестування на проникнення мобільного застосунку складається з декількох процесів, кожен з яких має власне функціональне призначення. Схема процесів інформаційної технології наведена на рис. 2.1.

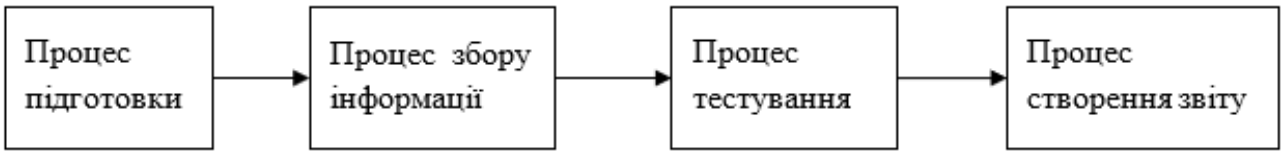


Рисунок 2.1 – Схема процесів технології на тестування проникнення

1) Процес підготовки. На даному етапі потрібно оцінити складність завдання, оцінити необхідні ресурси.

Процес включає в себе такі етапи (рис. 2.2).

– пошук кількість годин на аналіз вихідного коду;

– аналіз кількісті спеціалістів, яких необхідно задіяти у підготовчих роботах;

– аналіз вартості підготовчих робіт.

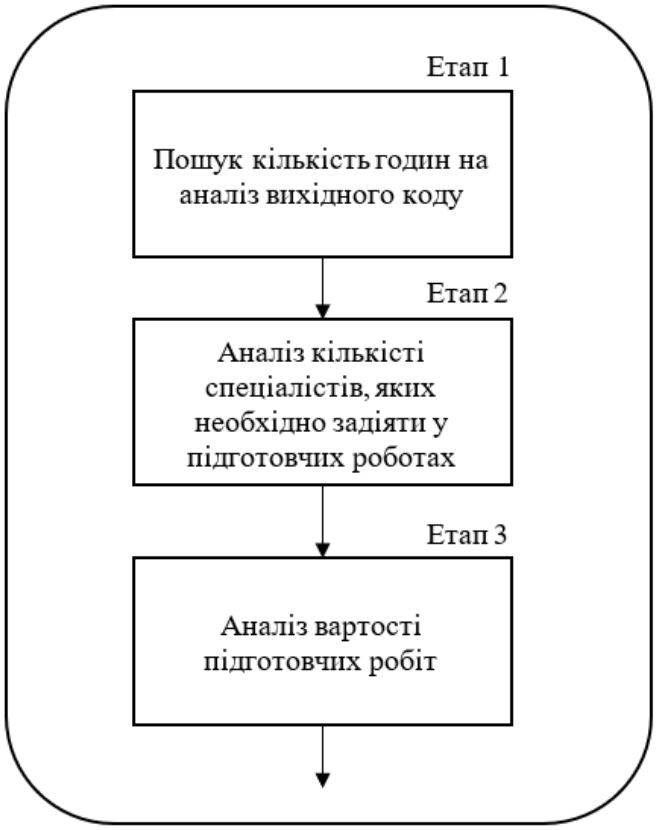


Рисунок 2.2 – Схема процесу підготовки

Вказаний процес є обов’язковим при тестуванні мобільного застосунку на проникнення. Модель процесу можна описати як:

$$I = <T_{prep}, N_{specialists}, C_{prep}>$$

де, T_{prep} - кількість годин на аналіз вихідного коду, $N_{specialists}$ - кількість спеціалістів, C_{prep} - вартість підготовчих робіт.

2) Процес збору інформації. Сканування додатка на вразливості, аналіз результатів сканування, збір інформації.

Процес включає в себе такі етапи (рис. 2.3).

– пошук потенційних вразливостей;
– аналіз знайдених вразливостей.

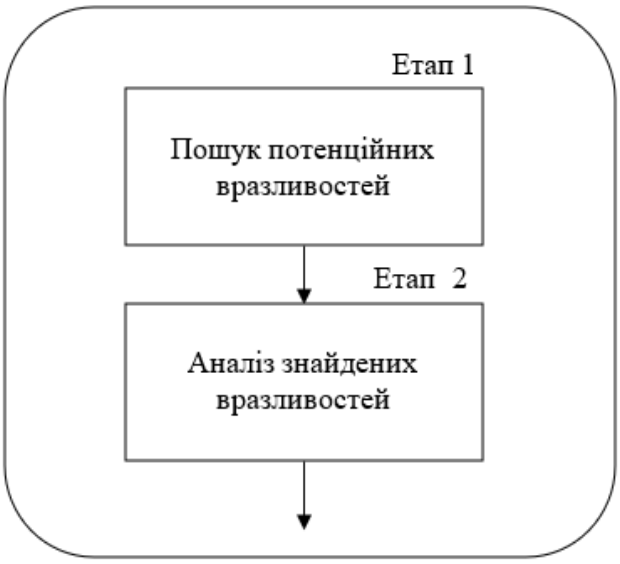


Рисунок 2.3 – Схема процесу збору інформації

На етапі пошук потенційних вразливостей відбуваються наступні етапи:

– Збір інформації про архітектуру додатка: це включає аналіз коду, дослідження використовуваних бібліотек та фреймворків.

– Виявлення вразливостей в реалізації безпеки: це може охоплювати перевірку наявності недостатньої аутентифікації, можливість введення некоректних даних, використання застарілих методів шифрування та інше.

– Аналіз наявних слабких місць у застосунку: дослідження можливих проблем, пов'язаних з використанням пам'яті, обробкою даних, мережевими протоколами та іншими аспектами програмного забезпечення.

На етапі аналізу знайдених вразливостей збирається наступна інформація:

– Опис виявлених вразливостей: детальний опис кожної знайденої уразливості, включаючи тип, місце розташування, можливий вплив на систему та шляхи її виправлення.

– Оцінка ризиків: визначення ступеня загрози, яку вона представляє для безпеки системи, та оцінка потенційних наслідків в разі експлуатації цих вразливостей зловмисниками.

– Рекомендації щодо виправлення: розробка конкретних рекомендацій та стратегій для виправлення знайдених вразливостей, включаючи практичні кроки та методи, які можна використовувати для забезпечення більшої безпеки

застосунку.

Вказаний процес є обов’язковим при тестуванні мобільного застосунку на проникнення.

Модель процесу можна описати як:

$$R_{data} = < T_{set}, I_{set}, T_{app} >$$

де, R_{data} – набір виявлених вразливостей, T_{set} – набір технології якими буде тестуватися мобільний застосунок, наприклад такі як: OWASP, NIST SP 800-163, MAST, MITRE ATT&CK і т.п., I_{set} – набір інструментів які будуть використовуватися для тестування мобільного застосунку, наприклад такі як: OWASP ZAP, MobSF, Drozer, Frida і т.п. та T_{app} – тип застосунку, до типу застосунку можна віднести технології які використовує застосунок наприклад такі як: Rest API, local Database, gRPC і т.п.

3) Процес тестування. Виконання атак на застосунок, перевірка на вразливості, аналіз результатів тестування. Процес включає в себе такі етапи (рис. 2.4).

- підбір векторів атак та необхідних інструментів;
- тестування застосунку на вразливості;
- аналіз результату атак та виявлених вразливостей.

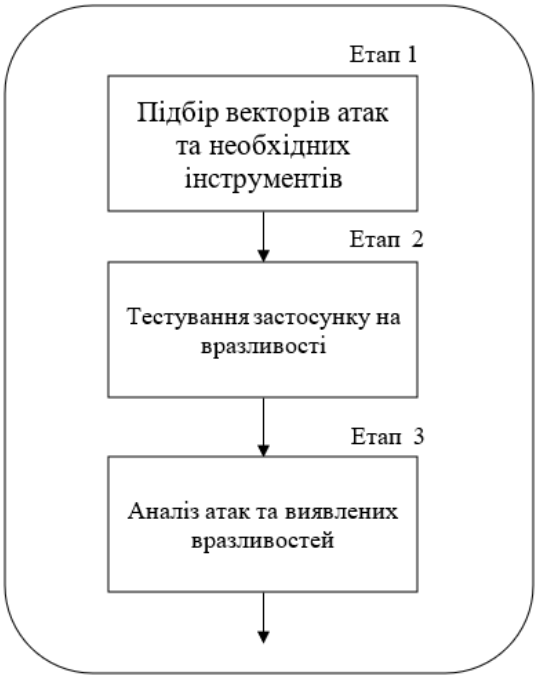


Рисунок 2.4 – Схема процесу тестування

На етапі підбору векторів атак та необхідних інструментів відбуваються наступні дії:

– На основі зібраної інформації та виявлених потенційних вразливостей на попередньому кроці ми вибираємо необхідний набір векторів атак та інструментів. Наприклад, для застосунку який не працює з локальною база даних, то для тестування не потрібно робити атаки на базу даних. Для різних типів застосунків потрібно використовувати різні типи атак та різні інструменти. Далі відбувається процес тестування застосунку на вразливості.

На етапі тестування відбувається тестування на проникнення мобільного застосунку.

– Після зібраної інформації щодо застосунку ми будемо робити наступні тест: тестування шифрування даних у мережі, перевірка конфігурації криптографічних стандартних алгоритмів, перевірка локального сховища на конфіденційні дані, перевірка конфіденційних даних, тестування резервних копій на конфіденційні дані, тестування дозволів програми, тестування локального сховища для перевірки введених даних.

– Деталі кожного етапу: опис процесу виконання кожного етапу тестування, виконання тестування на вразливості та оцінку результатів.

– Використані інструменти та методи: перелік використаних інструментів та методів для виконання атак та оцінки вразливостей.

На етапі аналізі атак та виявлених вразливостей описується:

– Деталі виявлених вразливостей: опис виявлених вразливостей, включаючи тип, місце розташування та можливий вплив на систему.

– Деталі виявлених атак: оцінка результатів кожної атаки, їх успішність та можливий вплив на функціональність застосунку.

Вказаний процес є обов’язковим при тестуванні мобільного застосунку на проникнення.

Модель процесу можна описати як:

$$A_t = \{ R_{data}, V_a \}$$

де A_t – результат тестування, R_{data} – набір виявлених вразливостей та V_a – вектори атаки.

Вектори атак можна описати як:

$$V_t = \{T_{1,2...N}, I_{1,2...N}\},$$

де $T_{1,2...N}$ – набір тестів та $I_{1,2...N}$ – набір інструментів.

4) Процес створення звіту. Складання звіту про виявлені вразливості та рекомендації по їх нейтралізації. Схему процесу наведено на рисунку 2.5.

- аналіз виявлених проблем;
- оцінка ризиків та встановлення пріоритетів;
- форматування звіту.

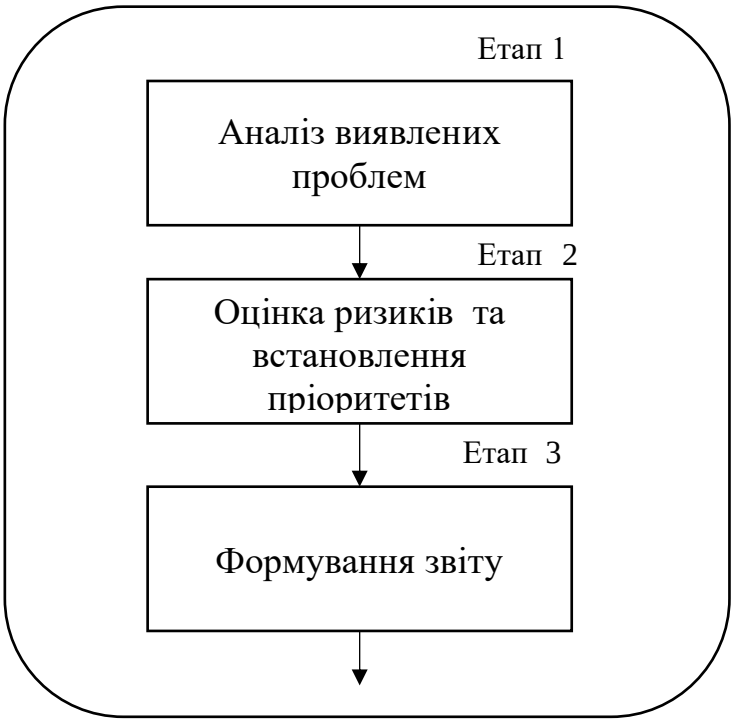


Рисунок 2.5 – Схема процесу створення звіту

На першому етапів відбувається:

– Опис знайдених вразливостей: детальний опис кожної знайденої уразливості, включаючи їх тип, можливий вплив на систему та шляхи експлуатації.

– Рекомендації щодо виправлення: конкретні пропозиції та рекомендації щодо виправлення кожної виявленої проблеми, включаючи практичні кроки та методи, які можна використовувати для забезпечення більшої безпеки.

На другому етапів відбувається:

– Оцінка ризиків: оцінка ступеня загрози, яку виявлені проблеми представляють для безпеки системи, та оцінка можливих наслідків в разі експлуатації цих вразливостей.

– Порядок пріоритетів: визначення порядку пріоритетів щодо виправлення виявлених проблем, виходячи з їх важливості та можливого впливу на діяльність застосунку.

На третьому етапів відбувається:

– Створення структура звіту: опис структури звіту, включаючи заголовки, розділи та підрозділи, що допомагають у логічній організації інформації.

– Форматування звіту: деталі форматування, такі як використання таблиць, графіків, списків та інших елементів, що полегшують сприйняття інформації.

Модель процесу можна описати як:

$$T_{result} = \{ I_{vulnerabilities}, R_{assessment}, R_{neutralizing}\}$$

де, T_{result} - результат створення звіту, $I_{vulnerabilities}$ - виявлені вразливості, $R_{assessment}$ - оцінка критичності, $R_{neutralizing}$ - набір рекомендацій по нейтралізації вразливостей.

В результаті створено моделі та схеми процесів інформаційної технології тестування на проникнення мобільного застосунку. Наступним кроком є розробка алгоритму тестування мобільного застосунку на проникнення для реалізації інформаційної технології.

2.2 Розробка алгоритму тестування мобільного застосунку на проникнення

Алгоритм тестування мобільного застосунку на проникнення складається із таких етапів (рис. 2.6):

- етап підготовки;
- етап збору інформації;

– етап тестування мобільного застосунку на проникнення;
– етап аналізу результатів та процес створення звіту.

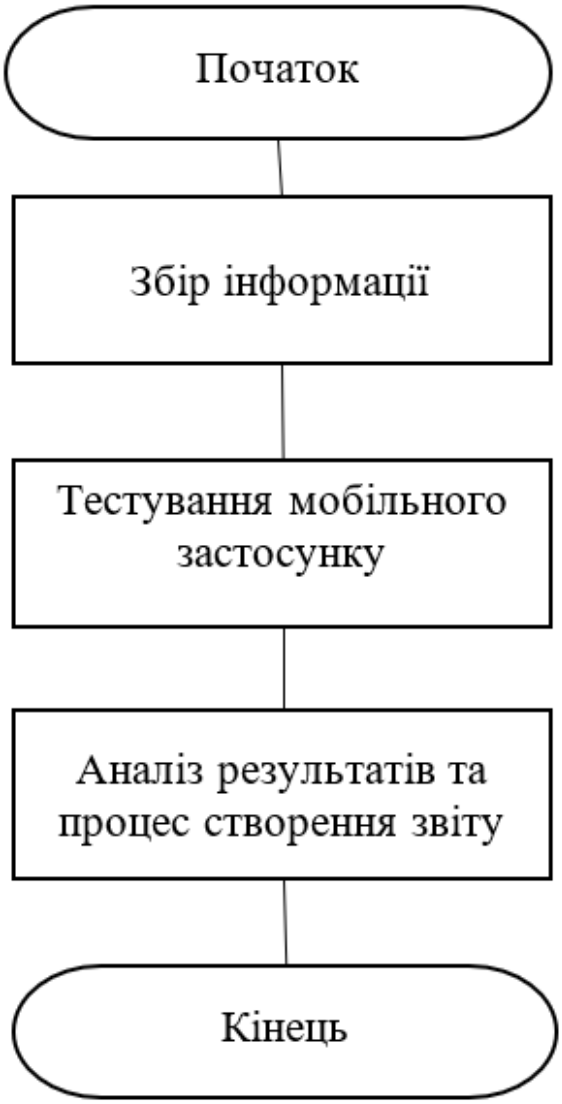


Рисунок 2.6 – Загальна схема процесу тестування мобільного застосунку на проникнення

Наступним етапом після створення схеми процесу тестування мобільного застосунку на проникнення, є етап збору інформації.

2.2.1 Етап збору інформації

Основна мета цього етапу полягає в зборі всієї необхідної інформації, що стосується мобільного застосунку, його архітектури, дизайну та безпеки (рис 2.7).

Рисунок 2.7 – Схема алгоритму збору інформації

Першим чином потрібно розпочати з огляду всієї документації, пов'язаної з мобільним застосунком. Це включає технічні специфікації, описи функціональності, вимоги до безпеки, архітектуру системи тощо.

Та як тестування відбувається за принципом «чорної та білої скриньки» ми маємо вихідний код застосунку. Потрібно проаналізувати вихідний код, провести його аналіз для виявлення можливих слабких місць та вразливостей.

Щоб переконатися, що всі вимоги до безпеки були правильно впроваджені у вихідному коді. Вихідний код можна проаналізувати за допомогою інструментів статичного аналізу, такі як SonarQube[18], Checkmarx [19], Fortify [20], а також інтегровані середовища розробки (IDE), наприклад,

Android Studio [23], IntelliJ IDEA [24]. Для перевірки аналізу коду надаю перевагу Android Studio [23] та IntelliJ IDEA [24] через їх простоту в використанні та безкоштовний доступ.

Наступним кроком потрібно перевірити всі конфігураційні файли, щоб переконатися, що налаштування безпеки та інші параметри налаштовані належним чином. Для перевірки конфігураційних файлів, такі як ConfigCheck [21], або вбудовані засоби в середовища розробки. Основною перевагою ConfigChecker є його здатність отримувати параметри конфігурації з різних джерел. Наприклад, він може читати або перевіряти значення:

- звичайні текстові файли;
- файли xml;
- файли властивостей java;
- файли маніфесту;
- java ini-файли windows;
- реєстр windows;
- файли конфігурації apache;
- файли ldif.

Після збору інформації потрібно проаналізувати наявні заходи безпеки, які вже впроваджені у мобільний застосунок, та оцінити їх ефективність.

При аналізі потрібно визначити можливі загрози в безпеці, які можуть виникнути при експлуатації мобільного застосунку. Це включає загрози, пов'язані зі зловживанням, витоками даних, атаками на мережу, неавторизованим доступом тощо.

2.2.2 Етап тестування мобільного застосунку

Проведення тестування мобільного застосунку на проникнення включає в себе ряд етапів та методів, які допомагають виявляти потенційні вразливості та забезпечувати високий рівень безпеки. До етапів тестування мобільного застосунку входять, такі кроки як (рис. 2.8):

Рисунок 2.8 – Схема алгоритму тестування мобільного застосунку

1) Аналіз вихідного коду застосунку: потрібно виконати такі дії: перевірка локального сховища на конфіденційні дані; перевірка на розкриття конфіденційних даних через інтерфейс користувача; пошук конфіденційної інформації на автоматично згенерованих знімках екрана; тестування локального сховища для перевірки введених даних; тестування дозволів програми; тестування резервних копій на конфіденційні дані. Це все можна зробити за допомогою Android Studio[23] та IntelliJ IDEA[24] через їх простоту в використанні та безкоштовний доступ.

2) Статичне тестування застосунку: Потрібно дослідити структуру та функціональність мобільного застосунку з метою ідентифікації потенційних слабкостей. Для аналізу мобільного застосунку потрібно використовувати такі інструменти аналізу як MobSF (Mobile Security Framework) та AndroBugs. Якщо порівнювати ці інструментами, то MobSF: має розширений функціонал для аналізу безпеки мобільних застосунків; включає в себе функції, такі як сканування вразливостей, тестування API, аналіз дозволів і безліч інших; підтримує як Android, так і iOS, що робить його універсальним інструментом для

аналізу мобільних застосунків; є активна спільнота користувачів і постійні оновлення так як AndroBugs вже не отримував стільки оновлень порівняно з MobSF. Кращим інструментом є Mobile Security Framework (MobSF).

3) Динамічне тестування:

– Для динамічного аналізу застосунків потрібно використовувати Frida[27] та Drozer [28]. Якщо потрібна гнучкість та можливість внедрювати власний код: Потрібно використовувати Frida. Він дозволяє перехоплювати та змінювати динамічну поведінку застосунків. Якщо потрібно інструмент для аудиту та виявлення вразливостей: Використовуйте Drozer. Він забезпечує сканування на вразливості та тестування на проникнення, але не має такої гнучкості, як Frida. Іноді найкраще використовувати обидва інструменти. Наприклад, сканування вразливостей з використанням Drozer, а потім використання Frida для подальшого аналізу та тестування застосунку під час його виконання.

– Тестування на вразливості міжкомпонентного взаємодії. Для даного тестування потрібно використовувати засоби перехоплення трафіку, такі як Burp Suite[25] та Wireshark[26]. До плюсів Burp Suite можна віднести: перехоплення та модифікація HTTP-запитів та відповідей, а також виявляти вразливості; ідеальний для тестування безпеки веб-застосунків, але не так потужний для загального аналізу всього мережевого трафіку. В сою чергу Wireshark можне: перехоплювати і аналізувати будь-який мережевий трафік, включаючи HTTP, TCP, UDP, і т.д; дозволяє докладно аналізувати всі аспекти мережевого взаємодії, що робить його корисним для різних задач, включаючи аналіз мережевого протоколу та розв'язання проблем зв'язку. Нам потрібно аналізувати загальний мережевий трафік та наша задача, є не пов'язаною із веб-додатками, Wireshark може бути більш відповідним інструментом.

2.2.3 Етап аналізу результатів та процес створення звіту

Даний етап є вирішальним у процесі тестування мобільних застосунків на проникнення, оскільки він допомагає зрозуміти серйозність виявлених вразливостей та визначити кроки для їх виправлення.

До аналізу результатів тестування мобільного застосунку на проникнення можна віднести, такі кроки як:

- Аналіз виявлених вразливостей. Виявлені вразливості повинні бути детально проаналізовані з точки зору їхнього потенційного впливу на безпеку застосунку та даних користувачів. Розуміння причин виникнення вразливостей є ключовим для подальшого виправлення.
- Оцінка серйозності вразливостей. Кожна вразливість повинна бути оцінена з точки зору потенційного впливу на безпеку системи. Серйозність вразливостей допомагає визначити порядок пріоритету для виправлення.

Фактори ризику «Risk Factors»: Кожному результату присвоюється два фактори для вимірювання його ризику. Фактори вимірюються за шкалою від 1 (Низький «low»), 2 (Середній «medium») до 3 (Високий «high»).

Вплив «Impact»: Вказує на вплив виявленого фактору на технічні та бізнес-операції. Він охоплює такі аспекти, як як конфіденційність, цілісність і доступність даних або систем, а також фінансові або фінансові або репутаційні втрати.

Ймовірність «Likelihood»: Вказує на потенційну можливість використання результатів дослідження. Вона враховує такі аспекти, як рівень кваліфікації зловмисника рівень кваліфікації зловмисника та відносна легкість використання.

Низький «Low»: Вразливості, які мають або дуже низький вплив на бізнес, або максимально високу ймовірність, або дуже дуже низьку ймовірність, але максимально високий вплив на бізнес, вважаються вразливостями з низьким ступенем серйозності. Крім того, вразливості, для яких і вплив, і ймовірність є низькими, вважаються низькими серйозність. Оцінка ризику від 0 до 3.

Середній «Medium»: Вразливості з помірним впливом на бізнес та ймовірністю вважаються серйозними "Medium". Сюди також відносяться

вразливості, які мають або дуже високий вплив на бізнес у поєднанні з низькою ймовірністю, або мають низький вплив на бізнес у поєднанні з дуже високою ймовірністю. Оцінка ризику від 3 до 6.

Високий «High»: Вразливості з високим або більшим впливом на бізнес і високою або більшою ймовірністю вважаються вразливими з високим ступенем серйозності. Оцінка ризику мінімум 6.

В таблиця 2.1 наведе приклад розрахунку «Оцінки ризиків». Оцінка ризику може бути визначена як добуток впливу та ймовірності. Наприклад, оцінка ризику обчислюється за формулою:

*Оцінка ризику = Вплив * Ймовірність.*

Таблиця 2.1 – Визначення факторів ризику

Фактор ризику	Вплив (Impact)	Ймовірність (Likelihood)	Оцінка ризику
Низька якість коду	2 (medium)	3 (high)	6 (medium)
Недостатня конфіденційність даних	3 (high)	2 (medium)	6 (medium)
Використання застарілих бібліотек	2 (low)	2 (medium)	4 (low)
Недостатня автентифікація користувачів	3 (high)	3 (high)	9 (high)

Процес створення звіту складається із таких кроків:

- Створення структури звіту. Звіт повинен мати чітку структуру, включаючи опис виявлених вразливостей, їх серйозність, можливі наслідки та рекомендації щодо виправлення.
- Детальний опис вразливостей. Кожна вразливість повинна бути детально описана, включаючи методи її експлуатації, можливі наслідки та рекомендації щодо виправлення. Потрібно додати приклади атак та відповідних заходів забезпечення.
- Рекомендації для виправлення. Потрібно надати конкретні рекомендації для виправлення кожної виявленої вразливості. Додати докладні кроки або рекомендації щодо вдосконалення коду, архітектури застосунку або поліпшення процесів безпеки.

- Висновки та рекомендації. В кінці потрібно сформувати висновки з аналізу та надати загальні рекомендації для подальших кроків з покращення безпеки застосунку.

Рисунок 2.9 – Схема алгоритму створення звіту

В таблиці 2.1 наведено матрицю відповідності інструментів тестування до етапів тестування.

Таблиця 2.1 – Матриця відповідності інструментів тестування до етапів тестування

Процеси / Інструменти	Підготовка	Збір інформації	Тестування	Створення звіту
Android Studio	+	+	–	+
Frida	–	+	+	+
Xposed	–	–	+	+
Drozer	+	+	+	+
MobXSS	–		+	+
MobSF	+	+	+	+
OWASP ZAP	–		+	+
Burp Suite Mobile Assistant	–	–	+	+
QARK	–	–	+	+
Andro-Bugs Framew-ork	–	–	+	+

Таблиця 2.2 – Матриця відповідності інструментів тестування до стандарту безпеки MASVS OWASP

MASVS / Інст-рументи	STORAGE		CRYPTO		AUTH			NETWORK		PLATFORM			CODE				RESILIENCE			
	1	2	1	2	1	2	3	1	2	1	2	3	1	2	3	4	1	2	3	4
Android Studio	+	+	+	+	+	+	+	–	–	–	+	+	+	+	+	+	+	–	–	+
Frida	+	–	–	–	–	–	–	–	–	–	+	–	–	–	–	+	–	+	+	+
Xposed	+	–	–	–	–	–	–	–	–	–	+	–	–	–	–	+	–	–	–	+
Drozer	+	+	–	–	–	–	–	–	–	–	+	+	–	–	–	–	+	–	+	+
MobXSS	+		–	–	–	–	–	–	–	–	+	–	–	–	–	–	–	–	–	–
MobSF	+	+	–	–	–	–	–	+	+	–	+	–	–	–	–	+	+	+	+	+
OWASP ZAP	–	–	–	–	–	–	–	+	+	+	+	–	–	–	–	–	–	–	–	–
Burp Suite Mobile Assistant	–	–	–	–	–	–	–	–	+	+	–	–	–	–	–	–	–	–	–	–
QARK	+	+	–	–	–	–	–	–	+	+	–	–	–	–	–	–	–	–	–	–
Andro-Bugs Framew-ork	+	–	–	–	–	–	–	–	+	–	+	–	–	–	–		+	–	–	–

Ці засоби тестування MASVS показують що можуть тестувати різні аспекти тестування безпеки мобільних застосунків. Залежно від конкретних потреб і платформ, які ви тестуєте, ви можете вибрати найбільш підходящі інструменти для тестування мобільних застосунків на проникнення.

На етапах підготовки та збору інформації потрібно визначити з якими технологіями додаток працює по типу Rest Api, gRPC, Local Database і т.п. для визначення наборів тестів.

У даному розділі було проаналізовано методи та засоби тестування мобільного застосунку на проникнення, було надано рекомендацій з у вигляді покрокових етапів тестування мобільного застосунку на проникнення.

3 РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ТЕСТУВАННЯ НА ПРОНИКНЕННЯ МОБІЛЬНОГО ЗАСТОСУНКУ

3.1 Тестування мобільного застосунку на проникнення

Для проведення тестування на проникнення за принципами «чорної скриньки» та «білої скриньки» було обрано Kali Linux [22] як тестове середовище та було обрано Android Studio [23] для аналізу вихідного коду застосунка. Тестування буде проводитися на комерційному застосунку який розміщений на Play Marker та має регулярне оновлення.

На етапі збору інформації було виявлено за допомогою Android Studio що над застосунком потрібно буде виконати наступні дії (табл. 3.1):

- перевірка конфігурації криптографічних стандартних алгоритмів;
- перевірка локального сховища на конфіденційні дані;
- тестування резервних копій на конфіденційні дані;
- тестування дозволів програми;
- тестування локального сховища для перевірки введених даних;
- тестування шифрування даних у мережі «REST API».

Таблиця 3.1 – Матриця відповідності інструментів тестування до знайдених вразливостей застосунку

Тести	Перевірка конфігурації криптографічних стандартних алгоритмів	Перевірка локального сховища на конфіденційні дані	Перевірка резервних копій на конфіденційні дані	Тестування дозволів програми	Тестування локального сховища для перевірки введених даних	Тестування шифрування даних у мережі «REST API».
MASVS Інструменти	CRYPTO-1	STORAGE-1	STORAGE-2	PLATFORM-1	STORAGE-1	NETWORK-1
Android Studio	+	+	+	+	+	+

Продовження табл. 3.1 – Матриця відповідності інструментів тестування до знайдених вразливостей застосунку

Тести / Інструменти	Перевірка конфігурації криптографічних стандартних алгоритмів	Перевірка локального сховища на конфіденційні дані	Перевірка резервних копій на конфіденційні дані	Тестування дозволів програми	Тестування локального сховища для перевірки введених даних	Тестування шифрування даних у мережі «REST API».
MobXSS	+	+	–	–	–	–
Frida	+	–	+	+	–	–
Xposed	–	–	+	–	–	–
Drozer	+	+	+	+	+	–
MobSF	–	+	+	–	–	+
OWASP ZAP	–	–	–	–	–	+
Burp Suite Mobile Assistant	–	–	–	–	–	+
QARK	+	–	+	+	–	–
Andro-Bugs Framew-ork	+	–	+	–	–	–

За допомогою Android Studio потрібно проаналізувати вихідний код комерційного застосунку та знайти потенційні вразливості.

Перевірка конфігурації криптографічних стандартних алгоритмів. Для цього потрібно використати Android Studio.

При аналізі не було ваиявлено криптографічних екземплярів. До екзамплярів відносяться:

- класи Cipher, Mac, MessageDigest, Signature;
- інтерфейси Key, PrivateKey, PublicKey, SecretKey;
- функції getInstance або generateKey;
- виключення KeyStoreException, CertificateException, NoSuchAlgorithmException;
- класи, що використовують пакети java.security.*, javax.crypto.*, android.security.* та android.security.keystore.*.

Для перевірки локального сховища на конфіденційні дані та локального сховища для перевірки введених даних потрібно використовувати Android Studio. При аналізі було виявлено що дані зберігаються в SharedPreferences.

– Зберігання даних відбувається за допомогою SharedPreferences (рис. 3.1). SharedPreferences API зазвичай використовується для постійного збереження невеликих колекцій пар ключ-значення. Дані, що зберігаються в об’єкті SharedPreferences, записуються у звичайний текстовий файл XML (рис. 3.2). Об’єкт SharedPreferences можна оголосити загальночитаним (доступним для всіх програм) або приватним, в нашому випадку SharedPreferences оголошений як приватний.

```
single { provideSharedPref(androidApplication()) }

fun provideSharedPref(app: Application): SharedPreferences {
return app.applicationContext.getSharedPreferences(
    SHARED_PREFERENCE_NAME,
    Context.MODE_PRIVATE
)
```

Рисунок 3.1 – Код реалізації SharedPreferences в приватному режимі

Дані, збережені в SharedPreferences, не шифруються за замовчуванням. Це означає, що якщо хтось отримає доступ до файлів пристрою, він може легко переглянути та змінити збережені дані. Якщо намагатися зберігати паролі користувача, використовуючи SharedPreferences, це не є безпечним. Паролі повинні бути захищені, і для цього краще використовувати більш безпечні механізми, такі як KeyStore чи бібліотеки керування аутентифікацією. Доступ до SharedPreferences може бути легко отриманий іншими додатками або навіть користувачем, якщо вони мають кореневані (rooted) пристрої.

SharedPreferences не надає механізм контролю доступу до даних якщо має тип доступу MODE_WORLD_READABLE «дає змогу іншим додаткам читати налаштування» або MODE_WORLD_WRITEABLE «дає змогу іншим додаткам записувати нові налаштування». Інші додатки можуть мати доступ до ваших

SharedPreferences, що може призвести до небажаного розкриття конфіденційної інформації.

Рисунок 3.2 – Зміст SharedPreferences файлу

Можна побачити що дані зберігаються у відкритому вигляді.

– При відкриті пункту меню Logcat в Android Studio, було знайдено логування конфіденційних даних, який виводить журнал системних повідомлень, включаючи повідомлення, які працюють з REST API (рис. 3.3).

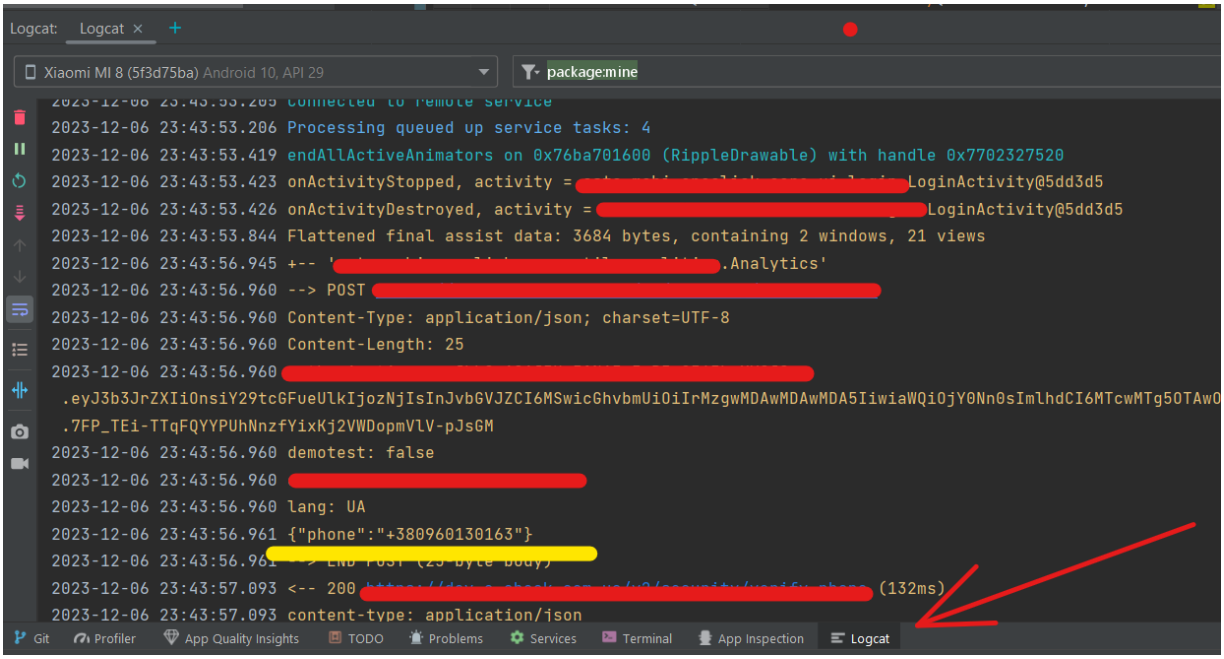


Рисунок 3.3 – Конфіденційні дані користувача

– Виявлено кешовані зображення в файловій системі застосунку за шляхом *data/app_name/cache/images* (рис. 3.4).

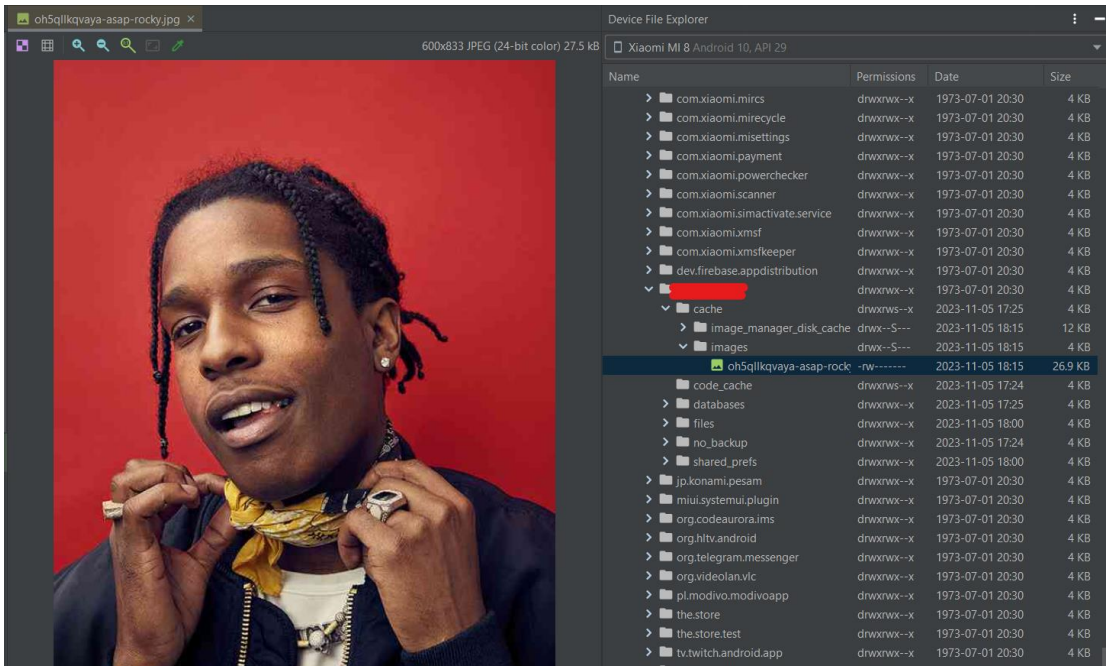


Рисунок 3.4 – Вигляд кешованого зображення в файлах застосунку

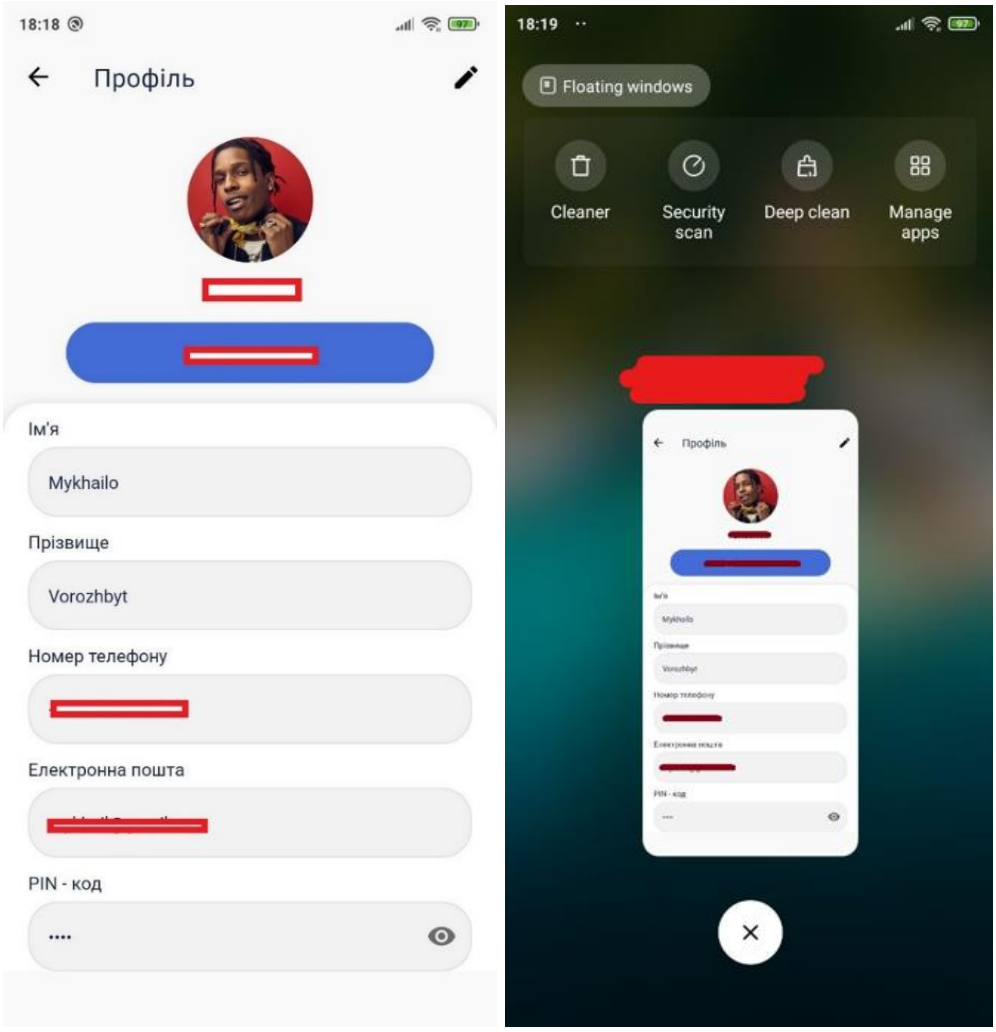
Перевірка резервних копій на конфіденційні дані. Для цього потрібно використати Android Studio.

– В файлі AndroidManifest.xml було знайдено allowBackup налаштування яке було дозволеному режимі (рис.3.5). Це налаштування дозволяє будь-кому робити резервну копію даних програми за допомогою adb. Це дозволить користувачам, які увімкнули налагодження по USB, копіювати дані програми з пристрою.

```
<manifest ... >
    ...
    <application android:allowBackup="true" ... >
        ...
    </application>
</manifest>
```

Рисунок 3.5 – Можливість повного резервного копіювання в Android

– Скріншот поточного екрану, коли програма Android переходить у фоновий режим, і відображається в списку відкритих застосунків, це може призвести до витоку конфіденційної інформації користувача. (рис. 3.6 (а), 3.6 (б)).

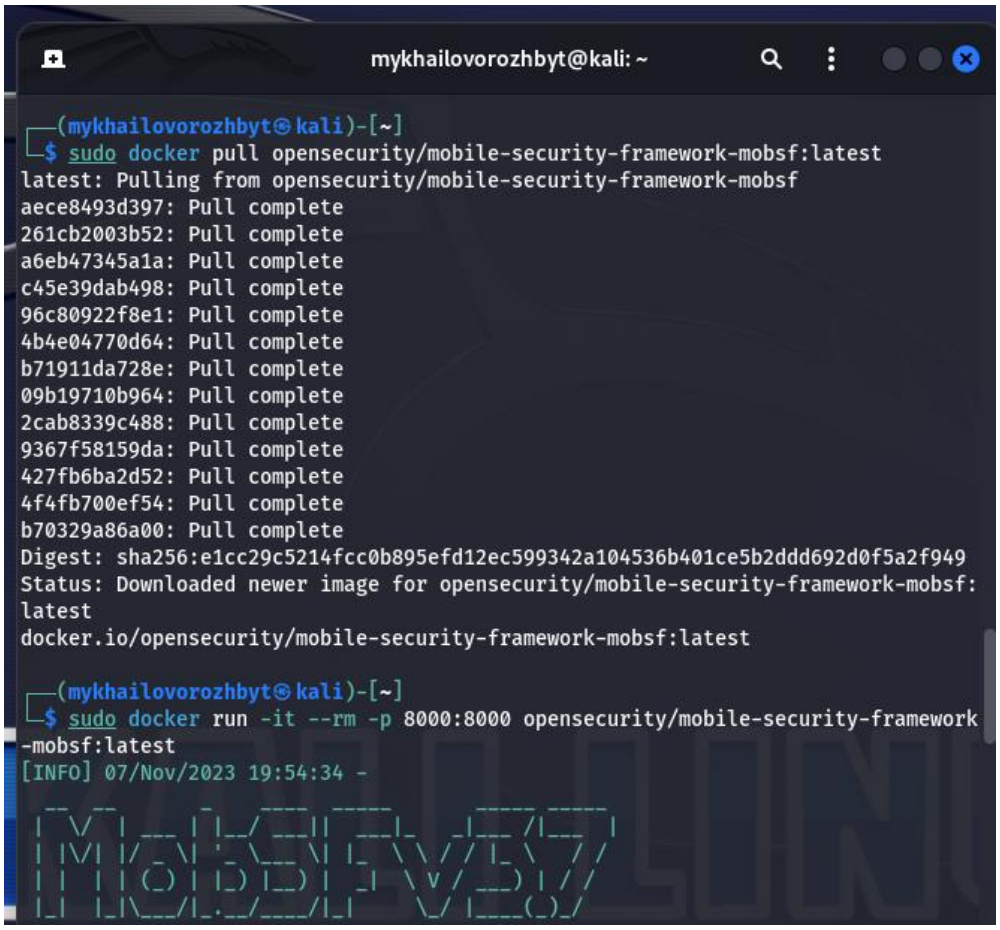


а) б)

Рисунки 3.6 – Екран з конференційними даними користувача: а – в застосунку, б – в згорнутому режимі

В основі статичного аналізу перш за все лежить отримання вихідного коду, двійкового коду, його вивчення для виявлення вразливостей, аналіз функціонування програми.

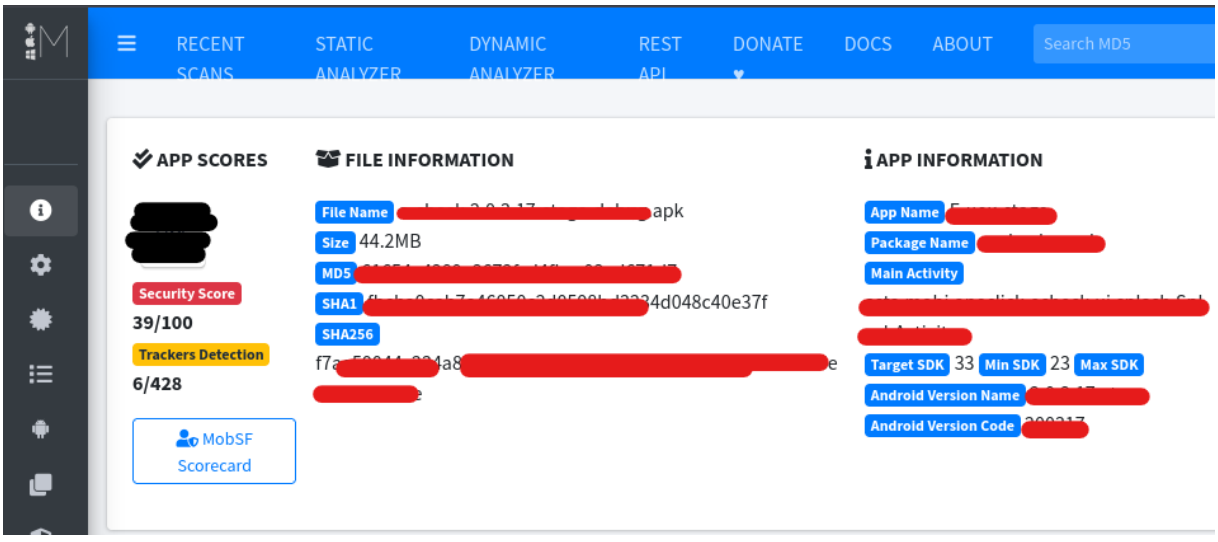
За допомогою утиліти MobSF (Mobile Security Framework) проводимо статичний аналіз нашого застосунку (рис. 3.7).



Рисунки 3.7 – Налаштування утиліти MobSF

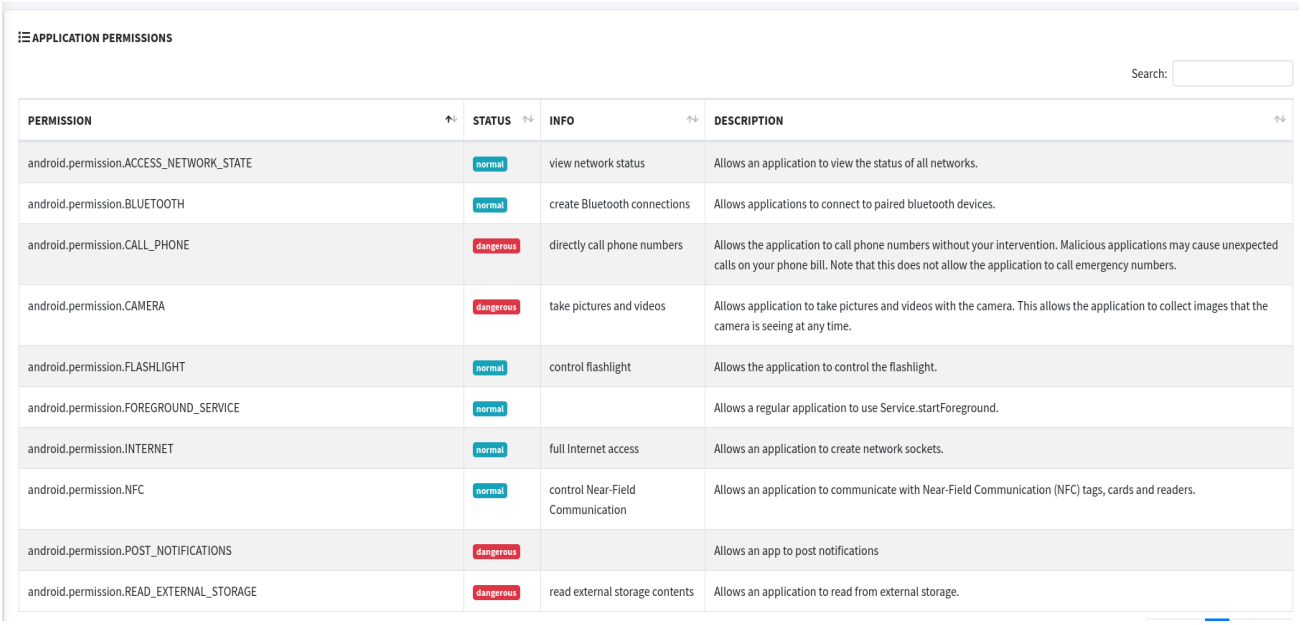
Після завантаження APK застосунку, MobSF надає наступну інформацію:

Інформація про оцінка безпеки, файлову інформацію та інформації про застосунок (рис. 3.8).



Рисунки 3.8 – Налаштування утиліти MobSF

Для перевірки та тестування дозволів програми потрібно використовувати утиліти MobSF. Виявлено різні типи дозволів які використає застосунок (рис. 3.9).



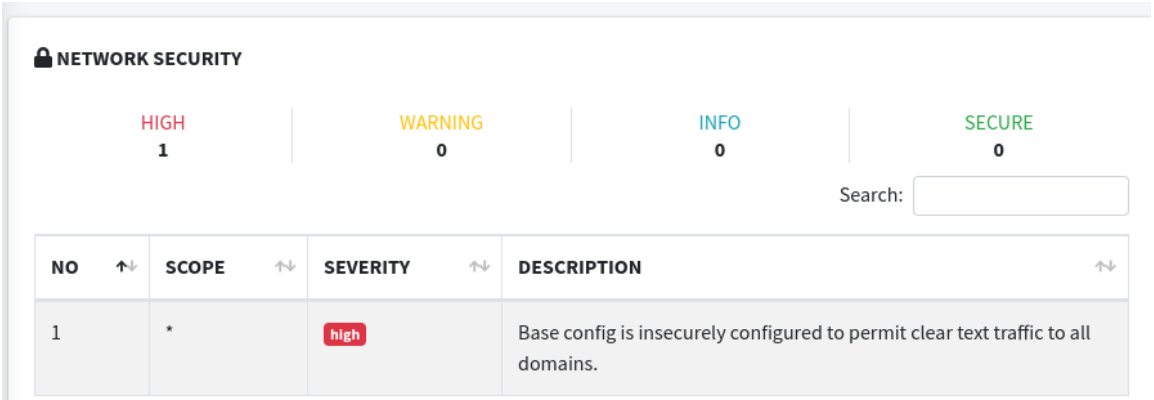
APPLICATION PERMISSIONS

Search:

PERMISSION	STATUS	INFO	DESCRIPTION
android.permission.ACCESS_NETWORK_STATE	normal	view network status	Allows an application to view the status of all networks.
android.permission.BLUETOOTH	normal	create Bluetooth connections	Allows applications to connect to paired bluetooth devices.
android.permission.CALL_PHONE	dangerous	directly call phone numbers	Allows the application to call phone numbers without your intervention. Malicious applications may cause unexpected calls on your phone bill. Note that this does not allow the application to call emergency numbers.
android.permission.CAMERA	dangerous	take pictures and videos	Allows application to take pictures and videos with the camera. This allows the application to collect images that the camera is seeing at any time.
android.permission.FLASHLIGHT	normal	control flashlight	Allows the application to control the flashlight.
android.permission.FOREGROUND_SERVICE	normal		Allows a regular application to use Service.startForeground.
android.permission.INTERNET	normal	full Internet access	Allows an application to create network sockets.
android.permission.NFC	normal	control Near-Field Communication	Allows an application to communicate with Near-Field Communication (NFC) tags, cards and readers.
android.permission.POST_NOTIFICATIONS	dangerous		Allows an app to post notifications
android.permission.READ_EXTERNAL_STORAGE	dangerous	read external storage contents	Allows an application to read from external storage.

Рисунки 3.9 – Дозволи застосунку

Для тестування шифрування даних у мережі «REST API» потрібно використовувати утиліти MobSF. Виявлено проблему конфігурації із шифруванням трафіку в мережі при використанні REST API (рис. 3.10). Опис помилки: Базову конфігурацію ненадійно налаштовано для дозволу трафіку відкритого тексту до всіх доменів.



NETWORK SECURITY

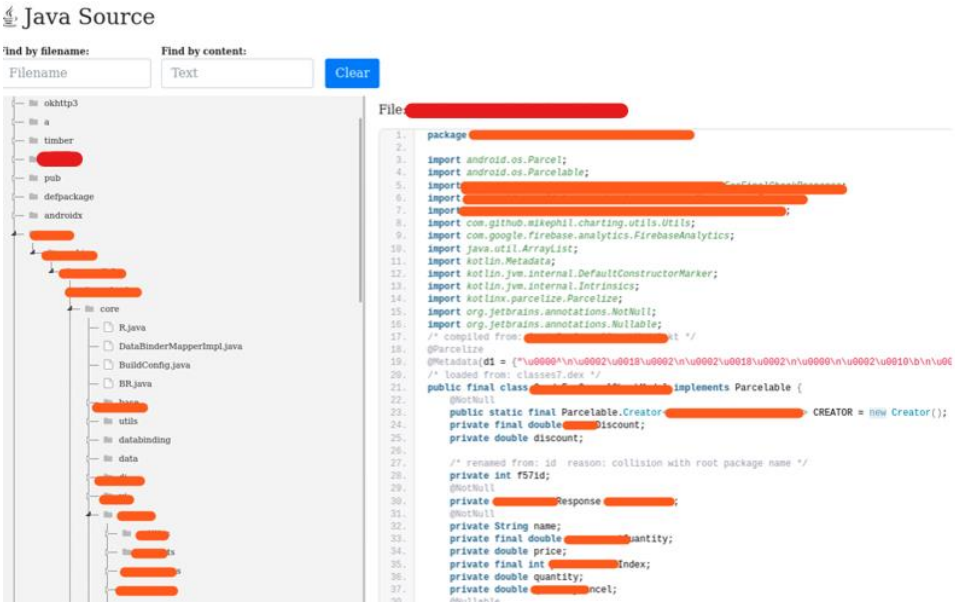
HIGH	WARNING	INFO	SECURE
1	0	0	0

Search:

NO	SCOPE	SEVERITY	DESCRIPTION
1	*	high	Base config is insecurely configured to permit clear text traffic to all domains.

Рисунки 3.10 – Шифрування даних у мережі

Виявлено що Застосунок без належного обфускування коду. Декомпільований Застосунок показує всі імена класів та параметрів (рис. 3.11).



Рисунки 3.11 – Клас без належної обфускації

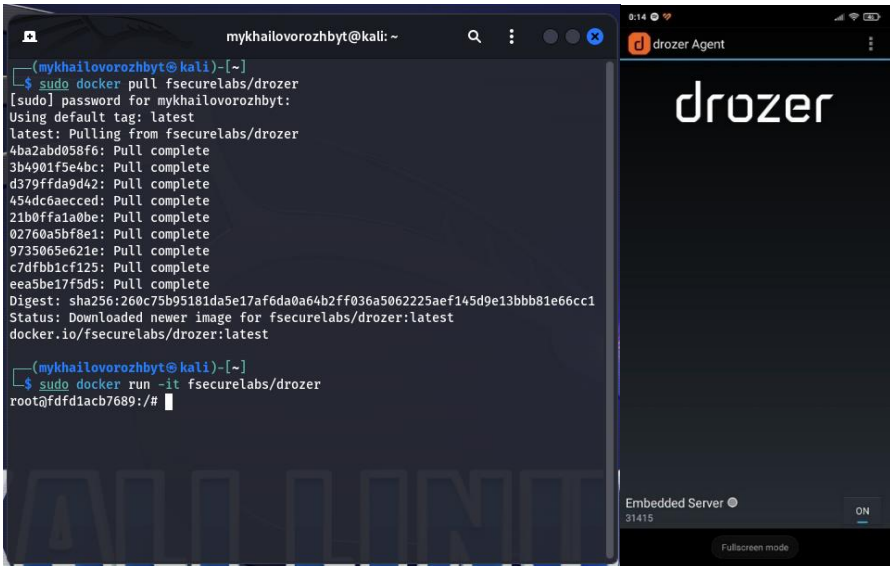
За допомогою Kali Linux використовуючи утиліти drozer (рис. 3.11 (а)) та drozer agent (рис. 3.11 (б)) будемо тестувати застосунок на проникнення:

Перше потрібно зайти в drozer консоль та налаштувати drozer agent такими командами:

sudo docker pull fsecurelabs/drozer

sudo docker run -it fsecurelabs/drozer

sudo adb forward tcp:31415 tcp:31415



а) б)

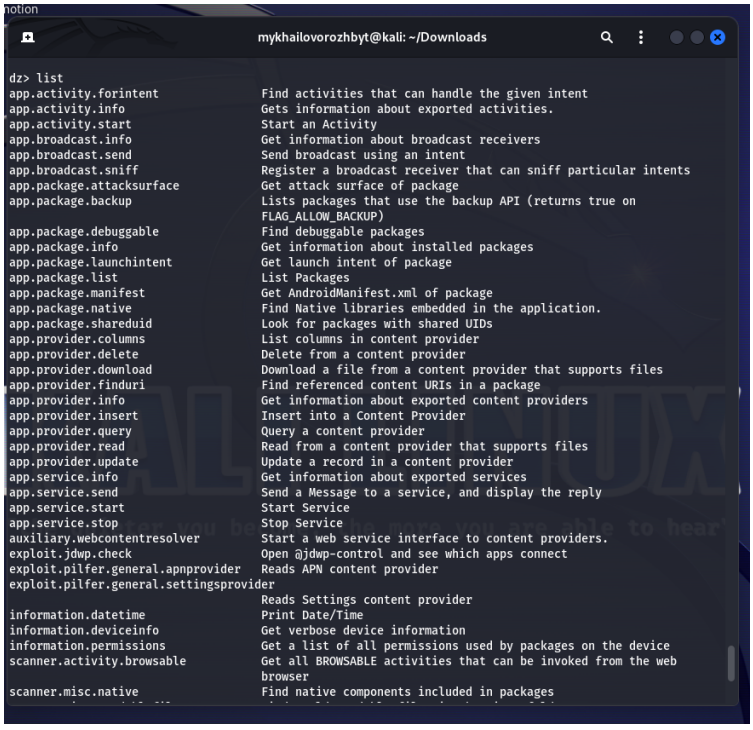
Рисунки 3.12 – Результат налаштування drozer agent: а - в kali linux, б - в смартфоні

Під’єднали drozer agent до нашого смартфону (рис. 3.13) за допомогою команди:

drozer console connect --server 192.168.0.173

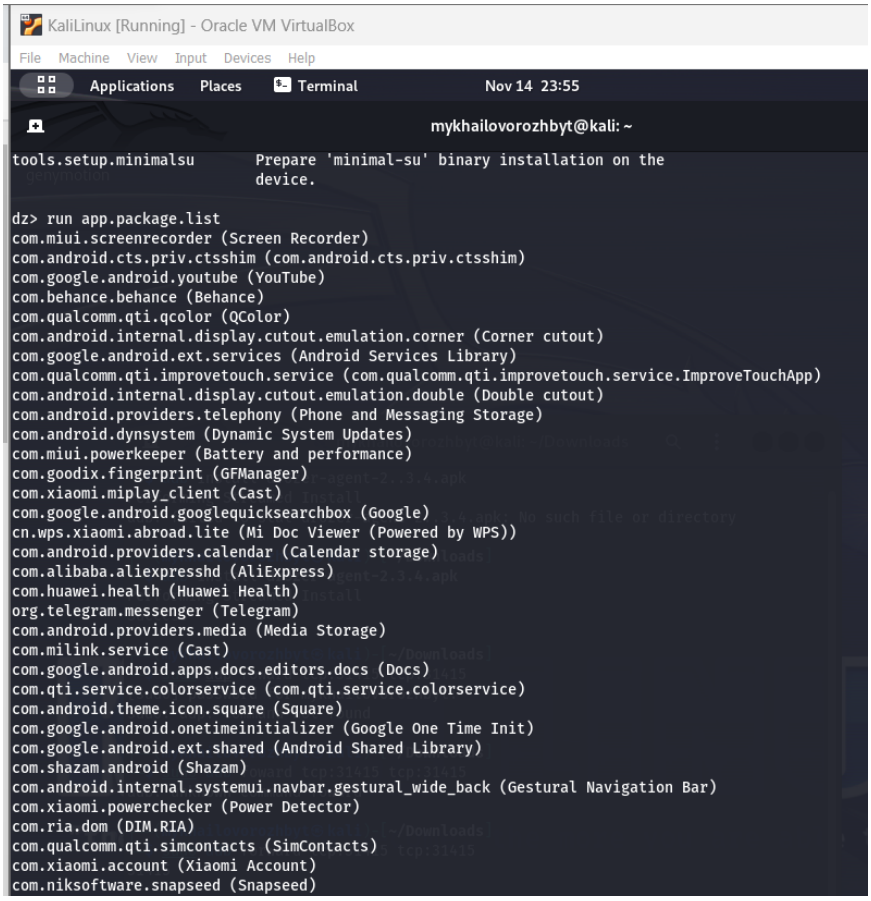
Рисунки 3.13 – Результат підключення drozer agent до смартфону по IP

Команд list - показує список усіх модулів drozer, які можуть бути запущені у поточному сеансі. Приховує модулі, на запуск яких ви не маєте відповідних дозволів (рис. 3.13).



Рисунки 3.13 – Список модулів dorzer

За допомогою команди *run app.package.list*, в drozer консолі знаходимо всі пакети різних застосунків на смартфоні (рис. 3.14).

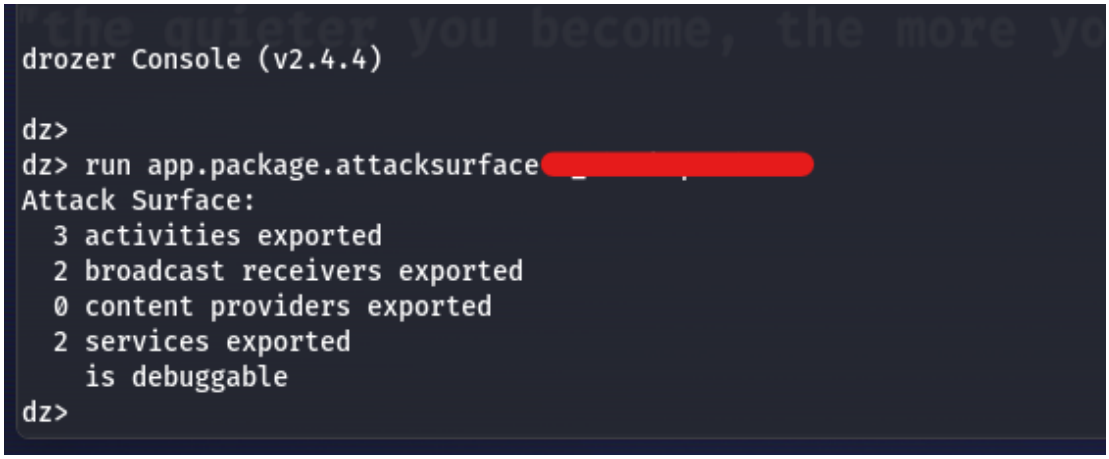


Рисунки 3.14 – Список всіх пакетів на смартфоні

За допомогою команди *run app.service.info* в drozer консолі отримуємо інформацію про експортовані послуги та за допомогою команди *run app.activity.start –component SplashActivity* запускаємо активність *SplashActivity* (рис. 3.15).

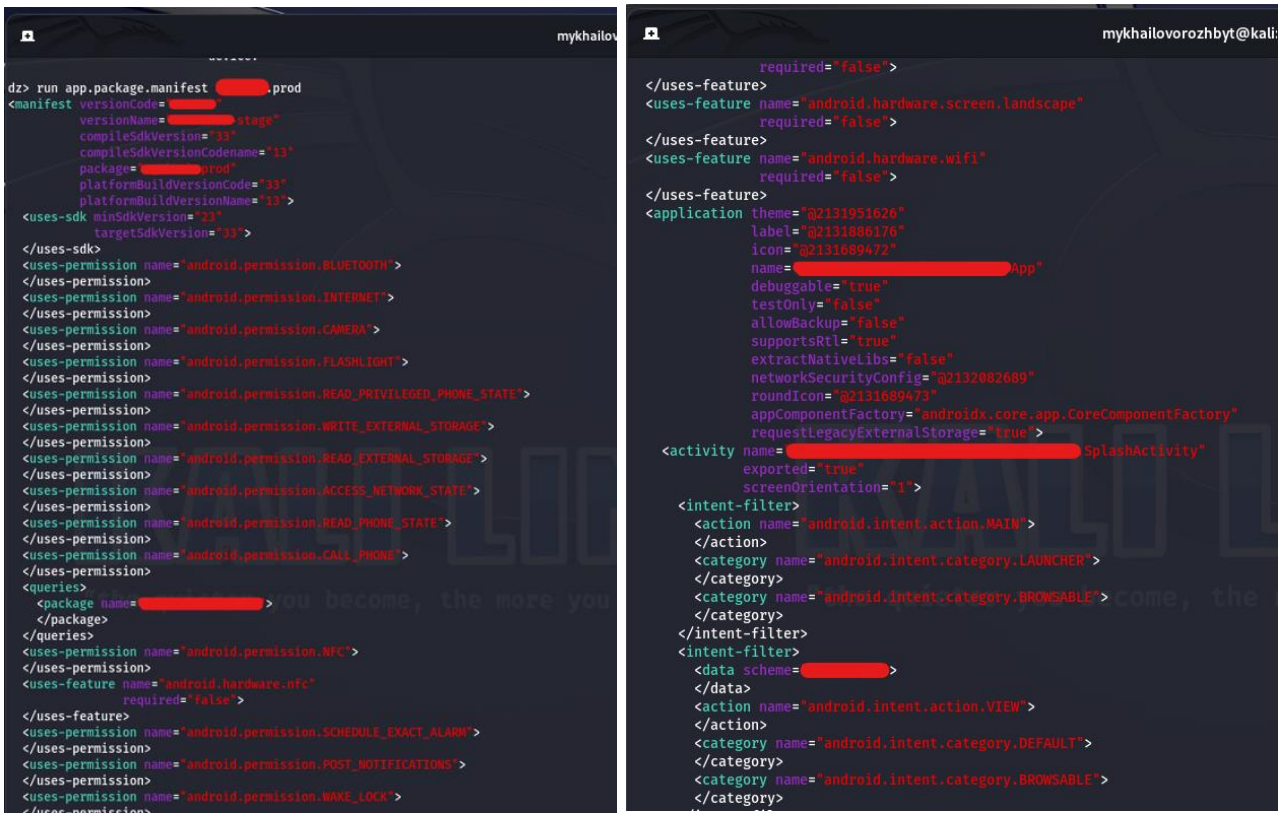
Рисунки 3.15 – Рузультат атаки на SplashActivity

За допомогою команди *run app.package.attacksurface* в drozer консолі отримуємо поверхню атаки пакета (рис 3.16)



Рисунки 3.16 – Пакети застосунку

За допомогою команди run app.package.manifest app.prod в drozer консолі отримуємо вміст файлу AndroidManifest.xml, на рисунку 3.17 (а) видно які дозволи використовує застосунок та на рисунку 3.17 (б) видного головне активність застосунку.



а) б)

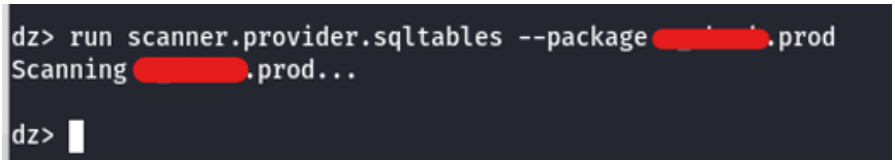
Рисунки 3.17 – Вміст файлу AndroidManifest.xml: а – 1 частина файлу, б – друга частина файлу

За допомогою команди *run scanner.provider.injection* в drozer консолі використовується для перевірки на наявність уразливостей ін'єкцій контент-провайдерів в Android-застосунку.

Рисунки 3.18 – Результат уразливостей ін'єкцій контент-провайдерів

Видно, що за допомогою даної атаки не було виявлено уразливостей ін'єкцій контент-провайдерів в Android-застосунку.

Команда *run scanner.provider.sqltables --package app.prod* в drozer консолі використовується для вилучення інформації про таблиці бази даних SQLite, відкриті постачальниками контенту в Android-застосунку.



Рисунки 3.19 – Результат атаки SQLite таблиць

Видно що атака була не успішною, SQLite таблиць було не виявлено.

Після тестування застосунку потрібно зробити аналіз тестування застосунку на проникнення та надати рекомендації з їх знешкодженням.

3.2 Аналіз результатів тестування мобільного застосунку на проникнення

При тестуванні мобільного застосунку на проникнення було виявлено такі вразливості:

1) Неправильне використання API SharedPreferences часто може призвести до розкриття конфіденційних даних. Для зберігання конфіденційних даних рекомендовано використовувати EncryptedSharedPreferences [29] та Android Keystore System [30].

EncryptedSharedPreferences - це свого роду обгортка для SharedPreferences. Крім того, вона виконує операції шифрування і розшифрування під час зберігання і отримання значень зі спільних налаштувань (рис. 3.20).

```
MasterKey masterKey = new MasterKey.Builder(context)
    .setKeyScheme(MasterKey.KeyScheme.AES256_GCM)
    .build();

SharedPreferences sharedPreferences = EncryptedSharedPreferences.create(
    context,
    "secret_shared_prefs",
    masterKey,
    EncryptedSharedPreferences.PrefKeyEncryptionScheme.AES256_SIV,
    EncryptedSharedPreferences.PrefValueEncryptionScheme.AES256_GCM
);

// use the shared preferences and editor as you normally would
SharedPreferences.Editor editor = sharedPreferences.edit();
```

Рисунок 3.20 – Код реалізації EncryptedSharedPreferences об’єкта

Система Android Keystore дозволяє зберігати криптографічні ключі в контейнері, щоб ускладнити їх вилучення з пристрою. Коли ключі знаходяться у сховищі, ви можете використовувати їх для криптографічних операцій, при цьому матеріал ключів залишається неекспортабельним. Крім того, система сховища дозволяє обмежити, коли і як можна використовувати ключі, наприклад, вимагати автентифікацію користувача для використання ключів або обмежити використання ключів лише у певних криптографічних режимах. Android Keystore System захищає ключовий матеріал від несанкціонованого використання двома способами.

По-перше, вона знижує ризик несанкціонованого використання ключового матеріалу ззовні пристрою Android, запобігаючи вилученню ключового матеріалу з процесів застосунків і з пристрою Android в цілому.

По-друге, система сховища ключів знижує ризик несанкціонованого використання ключових матеріалів всередині пристрою Android, змушуючи додатки визначати дозволені способи використання своїх ключів, а потім забезпечуючи дотримання цих обмежень поза процесами застосунків (рис. 3.20).

```
val kpg: KeyPairGenerator = KeyPairGenerator.getInstance(
        KeyProperties.KEY_ALGORITHM_EC,
        "AndroidKeyStore"
)
val parameterSpec: KeyGenParameterSpec = KeyGenParameterSpec.Builder(
        alias,
        KeyProperties.PURPOSE_SIGN or KeyProperties.PURPOSE_VERIFY
).run {
    setDigests(KeyProperties.DIGEST_SHA256, KeyProperties.DIGEST_SHA512)
    build()
}

kpg.initialize(parameterSpec)

val kp = kpg.generateKeyPair()
```

Рисунок 3. 21 – Код реалізації Android Keystore

2) При аналізі було виявлено, що логування API викликів де можна було виявити конференційні дані користувача. Для блокування логування рекомендується налаштувати вайл ProGuard (proguard-rules.pro), потрібного додати наступний рядок коду для блокування логування:

*-assumenosideeffects class android.util.Log { *; }*

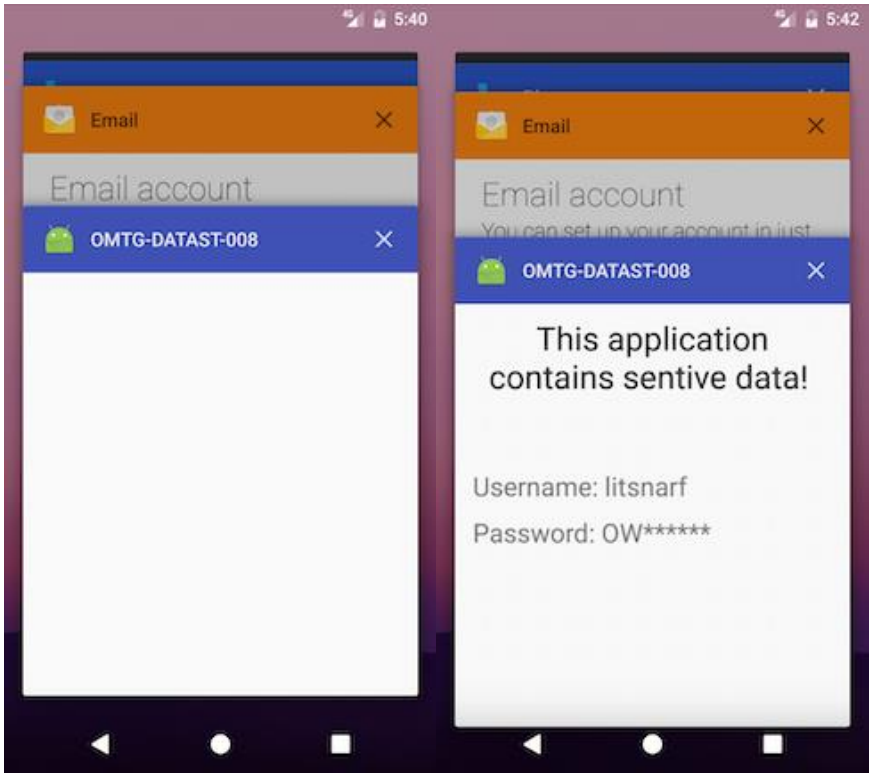
3) Часто для відображення для зображення на інтерфейсі користувача вико-ристовується бібліотека Glide [31]. Для запобігання кешування зображення потрібно додати до Glide параметри *diskCacheStrategy(DiskCacheStrategy.NONE)* та *skipMemoryCache(true)*, приклад ініціалізації Glide з даними параметрами можна побачити на рисунку 3. 22.

```
Glide.with(DemoActivity.this)
    .load(Uri.parse("file://" + imagePath))
    .diskCacheStrategy(DiskCacheStrategy.NONE)
    .skipMemoryCache(true)
    .into(mImage);
```

Рисунок 3.22 – Код реалізації Glide з увімкненим кешуванням

4) Android надає атрибут allowBackup для резервного копіювання всіх даних вашої програми. Цей атрибут встановлюється у файлі AndroidManifest.xml. Якщо значення цього атрибуту рівне true, пристрій дозволяє користувачам створювати резервні копії програми за допомогою Android Debug Bridge (ADB) за допомогою команди $ adb backup. Щоб заборонити резервне копіювання даних програми, встановіть атрибут *android:allowBackup* у значення false. Якщо цей атрибут недоступний, параметр allowBackup увімкнено за замовчуванням, і резервне копіювання потрібно вимкнути вручну *android:allowBackup=”false”*.

5) Під час тестування програми за допомогою чорної скриньки після переходу до екрану, який містить конфіденційну інформацію, і натискання кнопки «Додому», щоб перевести програму у фоновий режим. Як показано на рисунках 3.23, якщо FLAG_SECURE встановлено (рис. 3.23 (а)), знімок буде порожнім; якщо прапор не встановлено (рис. 3.3 (б)) буде видно конфіденційну інформацію користувача.



а) б)

Рисунок 3.23 – Застосунок з: а – увімкнений параметр FLAG_SECURE, б – вимкнений параметр FLAG_SECURE

Для блокування знімка екрана потрібно встановити параметр FLAG_SECURE (рис 3.24).

```
window.setFlags(WindowManager.LayoutParams.FLAG_SECURE,
                WindowManager.LayoutParams.FLAG_SECURE)

setContentView(R.layout.activity_main)
```

Рисунок 3.24 – Приклад застосування параметра FLAG_SECURE

6) Після тестуванням статичним аналізом було виявлено що застосунок без належного обфускування коду. Для обфусуквання коду рекомендується рекомендується налаштувати вайли ProGuard (proguard-rules.pro) та build.gradle.

До файлу proguard-rules.pro потрібно додати наступний код для обфускації всіх публічних та захищених класів:

*-keep class com.example.project** { public protected *; }*

Для файлу build.gradle потрібно ввімкнути параметри isMinifyEnabled та isShrinkResources.

– isMinifyEnabled (альтернатива назва minifyEnabled) цей параметр вмикає зменшення коду та обфускацію під час збірки релізної версії програми. Зменшення коду (code shrinking) видаляє невикористані класи, методи та ресурси, а обфускація робить код менш читабельним для зловмисників. Використання інструменту ProGuard для цих цілей дозволяє покращити продуктивність та зменшити розмір застосунку.

– isShrinkResources (альтернатива назва shrinkResources) цей параметр вмикає зменшення ресурсів (resource shrinking). Ресурси, які не використовуються в програмі, можуть бути вилучені під час збірки, що дозволяє зменшити розмір вихідного застосунку. Це особливо корисно для застосунків, які мають велику кількість ресурсів, але не використовують їх усі.

Після аналізу та надання рекомендацій кожної загрози потрібно розробити звіт з тестування мобільного застосунку на проникнення.

3.3 Створення звіту

Звіт повинен мати чітку структуру, включаючи опис виявлених вразливостей, їх серйозність, можливі наслідки та рекомендації щодо виправлення. Кожна вразливість повинна бути описана, включаючи методи її експлуатації, можливі наслідки та рекомендації щодо виправлення. Кожна вразливість повинна бути оцінена з точки зору потенційного впливу на безпеку системи. Серйозність вразливостей допомагає визначити порядок пріоритету для виправлення. Потрібно надати конкретні рекомендації для виправлення кожної виявленої вразливості. Додати докладні кроки або рекомендації щодо вдосконалення коду, архітектури застосунку або поліпшення процесів безпеки.

Зміст звіту про тестування на проникнення мобільного застосунку містить такі підрозділи:

- 1.0 Методологія розрахунку ризиків
- 2.0 Короткий зміст
- 2.1 Конфіденційні дані в журналах (Sensitive data in logs)
- 2.2 Відсутність належного обфускування коду застосунку
- 2.3 Автоматично згенеровані скріншоти дій з конференційними даними
- 2.4 Відсутність конфігурації відмовостійкості
- 3.0 Висновки

В розділі 1.0 описується методологія розрахунку ризиків.

Фактори ризику «Risk Factors»: Кожному результату присвоюється два фактори для вимірювання його ризику. Фактори вимірюються за шкалою від 1 (low), 2 (medium) до 3 (high).

Вплив «Impact»: Вказує на вплив виявленого фактору на технічні та бізнес-операції. Він охоплює такі аспекти, sяк конфіденційність, цілісність і доступність даних або систем, а також фінансові або фінансові або репутаційні втрати.

Ймовірність «Likelihood»: Вказує на потенційну можливість використання результатів дослідження. Вона враховує такі аспекти, як рівень

кваліфікації зловмисника рівень кваліфікації зловмисника та відносна легкість використання.

Високий «High»: Вразливості з високим або більшим впливом на бізнес і високою або більшою ймовірністю вважаються вразливими з високим ступенем серйозності. Оцінка ризику мінімум 6.

Середній «Medium»: Вразливості з помірним впливом на бізнес та ймовірністю вважаються серйозними "Medium". Сюди також відносяться вразливості, які мають або дуже високий вплив на бізнес у поєднанні з низькою ймовірністю, або мають низький вплив на бізнес у поєднанні з дуже високою ймовірністю. ймовірність. Оцінка ризику від 3 до 6.

Низький «Low»: Вразливості, які мають або дуже низький вплив на бізнес, або максимально високу ймовірність, або дуже дуже низьку ймовірність, але максимально високий вплив на бізнес, вважаються вразливостями з низьким ступенем серйозності. Крім того, вразливості, для яких і вплив, і ймовірність є низькими, вважаються низькими серйозність. Оцінка ризику від 0 до 3.

В розділі 2.0 описується Короткий зміст.

Проведено тест на проникнення в андроїд-Застосунок PROD_APP.

ІНФОРМАЦІЯ ПРО ЗАСТОСУНОК

Назва програми: PROD_APP

Назва пакету: prod

Основна діяльність: SplashActivity

Цільовий SDK: 33

Min SDK: 23

Назва версії для Android: 1.0.0.0.0-prod

Код версії Android: 100000

APK підписано

v1 signature: True

v2 signature: True

v3 signature: False

Ця оцінка "білої скриньки" та "чорної скрьки" була проведена для виявлення лазівок у застосунку з точки зору з точки зору безпеки. Ця оцінка була спрямована на виявлення вразливостей, присутніх в застосунку, які можуть призвести до ін'єкції, витоку інформації та інших ризиків, що можуть спричинити потенційних бізнес-збитків.

У цьому звіті представлені результати оцінки.

Загалом, під час оцінювання було виявлено декілька результатів, які будуть детально описані буде надано в розділі "Висновки".

В розділах 2.1 – 2.4 описується всі знайдені вразливості в мобільному застосунку.

В розділі 2.1 йде опис про конфіденційні дані в журналах.

Фактори ризику: Середній

Ймовірність: Середній

Вплив: Середній

Вектор атаки: Індивідуальний користувач

Опис: Застосунок використовує запис для реєстрації конфіденційну інформацію, таку як токен JWT користувача, дані користувача, ім'я та прізвище користувача (рис. 3.3).

Наслідки: У разі отримання зловмисником доступу до телефону (фізичного або віддаленого) він може встановити збір логів.

Рекомендація:

- Вимкніть збір конфіденційних даних.

Посилання: Sensitive Data - OWASP Mobile Application Security [32].

В розділі 2.1 описується відсутність належного обфускування коду застосунку.

Серйозність ризику: Низька

Ймовірність: Середня

Вплив: Низький

Вектор атаки: Індивідуальний користувач

Опис: Застосунок було скомпільовано без належного обфускації коду. У декомпільованому програма показує всі імена класів та параметрів. Обфускація коду може значно збільшити складність розуміння логіки роботи програми (рис. 3.11).

Вплив: Зловмисник може легко декомпілювати код, зрозуміти логіку програми та використати ці знання для подальших атак.

Рекомендація:

- Потрібно робити обфускацію коду перед компіляцією.

Посилання: Sensitive Data - OWASP Mobile Application Security [33].

В розділі 2.3 йде опис про автоматично згенеровані скріншоти дій з конференційними даними

Серйозність ризику: Низька

Ймовірність: Низька

Вплив: Низький

Вектор атаки: Індивідуальний користувач

Опис: Знімок екрана поточної активності робиться, коли програма Android переходить у фоновий режим, і відображається з естетичною метою, коли програма повертається на передній план. Однак це може призвести до витоку конфіденційної інформації.

Вплив: Зловмисник може отримати доступ до папки з автоматично згенерованими знімками екрана, що може призвести до розкриття інформації.

Рекомендація:

- Встановіть прапорець "БЕЗПЕЧНО" для дій з конфіденційною інформацією

Довідка: Finding Sensitive Information in Auto-Generated Screenshots - OWASP Mobile Application Security [34].

В розділі 2.4 йде опис про відсутність конфігурації відмовостійкості.

Серйозність ризику: Низька

Ймовірність: Низька

Вплив: Низький

Вектор атаки: Індивідуальний користувач

Опис: Програма не виявляє рутовані, емульовані або незашифровані пристрої.

Вплив: Зловмисник може використати цю ваду для тестування програми, щоб знайти слабкі місця або маніпулювати нею.

Рекомендація:

- Налаштувати виявлення кореневих, емульованих, незашифрованих пристроїв

Посилання: Testing Root Detection - OWASP Mobile Application Security [35], Testing Emulator Detection - OWASP Mobile Application Security [35].

В розділі 3.0 потрібно написати висновок з тестування мобільного застосунку на проникнення.

Під час проведення тесту на проникнення в андроїд застосунок було виявлено кілька критичних вразливостей, які можуть стати причиною серйозних ризиків для безпеки застосунку та конфіденційної інформації користувача.

4 ЕКОНОМІЧНА ЧАСТИНА

Науково-технічна розробка має право на існування та впровадження, якщо вона відповідає вимогам часу, як в напрямку науково-технічного прогресу та і в плані економіки. Тому для науково-дослідної роботи необхідно оцінювати економічну ефективність результатів виконаної роботи.

Магістерська кваліфікаційна робота на тему «Інформаційна технологія тестування на проникнення мобільного застосунку» відноситься до науково-технічних робіт, які орієнтовані на виведення на ринок (або рішення про виведення науково-технічної розробки на ринок можебути прийнято у процесі проведення самої роботи), тобто коли відбувається так звана комерціалізація науково-технічної розробки. Цей напрямок є пріоритетним, оскільки результатами розробки можуть користуватися інші споживачі, отримуючи при цьому певний економічний ефект. Але для цього потрібно знайти потенційного інвестора, який би взявся за реалізацію цього проекту і переконати його в економічній доцільності такого кроку.

Для наведеного випадку нами мають бути виконані такі етапи робіт:

1) проведено комерційний аудит науково-технічної розробки, тобто встановлення її науково-технічного рівня та комерційного потенціалу;

2) розраховано витрати на здійснення науково-технічної розробки;

3) розрахована економічна ефективність науково-технічної розробки у випадку її впровадження і комерціалізації потенційним інвестором і проведено обґрунтування економічної доцільності комерціалізації потенційним інвестором.

4.1 Проведення комерційного та технологічного аудиту науково-технічної розробки

Метою проведення комерційного і технологічного аудиту дослідження за темою «Інформаційна технологія тестування на проникнення мобільного застосунку» є оцінювання науково-технічного рівня та рівня комерційного

потенціалу розробки, створеної в результаті науково-технічної діяльності.

Оцінювання науково-технічного рівня розробки та її комерційного потенціалу рекомендується здійснювати із застосуванням 5-ти бальної системи оцінювання за 12-ма критеріями, наведеними в табл. 4.1 [37].

Таблиця 4.1 – Рекомендовані критерії оцінювання науково-технічного рівня і комерційного потенціалу розробки та бальна оцінка

Бали (за 5-ти бальною шкалою)					
	0	1	2	3	4
Технічна здійсненність концепції					
	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено працездатніс ть продукту в реальних
Ринкові переваги (недоліки)					
	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому
	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижчеза ціни
	Технічні та споживчі властивості продукту значно гірші, ніж в	Технічні та споживчі влас-тивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в	Технічні та споживчі властивості продукту значно кращі,
	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в
Ринкові перспективи					
	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великийринок з позитивною
	Активна конкуренція великих компаній на	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає

Для проведення технологічного аудиту залучені 3 незалежних експерти: керівник магістерської роботи доцент кафедри захисту інформації Вінницького національного технічного університету Куперштейн Л. М.; доцент кафедри захисту інформації Вінницького національного технічного університету Баришев Ю. В.; доцент кафедри захисту інформації Вінницького національного технічного університету Войтович О. П. Результати оцінювання науково-технічного рівня та комерційного потенціалу науково-технічної розробкипотрібно звести до таблиці.

Таблиця 4.2 – Результати оцінювання науково-технічного рівня і комерційного потенціалу розробки експертами

Критерії	Експерт (ПІБ, посада)		
	1	2	3
	Бали:		
1. Технічна здійсненність концепції	4	4	4
2. Ринкові переваги (наявність аналогів)	2	2	3
3. Ринкові переваги (ціна продукту)	3	3	3
4. Ринкові переваги (технічні властивості)	3	3	2
5. Ринкові переваги (експлуатаційні витрати)	2	2	2
6. Ринкові перспективи (розмір ринку)	2	2	2
7. Ринкові перспективи (конкуренція)	3	2	3
8. Практична здійсненність (наявність фахівців)	5	5	5
9. Практична здійсненність (наявність фінансів)	3	4	3
10. Практична здійсненність (необхідність нових матеріалів)	4	5	5
11. Практична здійсненність (термін реалізації)	4	4	4
12. Практична здійсненність (розробка документів)	4	3	4
Сума балів	39	39	40
Середньоарифметична сума балів $СБ_c$	39,3		

За результатами розрахунків, наведених в таблиці 4.2, зробимо висновок щодо науково-технічного рівня і рівня комерційного потенціалу розробки. При цьому використаємо рекомендації, наведені в табл. 4.3 [37].

Таблиця 4.3 – Науково-технічні рівні та комерційні потенціали розробки

Середньоарифметична сума балів СБ , розрахована на основі висновків експертів	Науково-технічний рівень та комерційний потенціал розробки
41…48	Високий
31…40	Вище середнього
21…30	Середній
11…20	Нижче середнього
0…10	Низький

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Інформаційна технологія тестування на проникнення мобільного застосунку» становить 39,3 бала, що, відповідно до таблиці 4.3, свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки вище середнього).

4.2 Розрахунок узагальненого коефіцієнта якості розробки

Окрім комерційного аудиту розробки доцільно також розглянути технічний рівень якості розробки, розглянувши її основні технічні показники. Ці показники по-різному впливають на загальну якість проектної розробки.

Узагальнений коефіцієнт якості (Вн) для нового технічного рішення розрахуємо за формулою [54]:

$$B_н = \sum \alpha_i \cdot \beta_i \, i=1$$

де k – кількість найбільш важливих технічних показників, які впливають на якість нового технічного рішення;

α_i – коефіцієнт, який враховує питому вагу і-го технічного показника в загальній якості розробки. Коефіцієнт αі визначається експертним шляхом і при цьому має виконуватись умова

$$\sum_{i=1}^{k} \alpha_i = 1;$$

β_i – відносне значення і-го технічного показника якості нової розробки.

Відносні значення β_i для різних випадків розраховуємо за такими формулами:

– для показників, зростання яких вказує на підвищення в лінійній залежності якості нової розробки:

$$\beta_i = \frac{I_{ні}}{I_{аі}},$$

де $I_{ні}$ та $I_{на}$ – чисельні значення конкретного і-го технічного показника якості відповідно для нової розробки та аналога;

– для показників, зростання яких вказує на погіршення в лінійній залежності якості нової розробки:

$$\beta_i = \frac{I_{аі}}{I_{ні}};$$

Використовуючи наведені залежності можемо проаналізувати та порівняти техніко-економічні характеристики аналогу та розробки на основі отриманих наявних та проектних показників, а результати порівняння зведемо до таблиці 4.4.

Таблиця 4.4 – Порівняння основних параметрів розробки та аналога.

Показники (параметри)	Одиниця вимірю-вання	Аналог	Проектований пристрій	Відношення параметрів нової розробки до аналога	Питома вага показника
Порівняльна швидкість передачі даних	бал	8	8,5	1,06	0,1
Надійність роботи мережі	%	90	95	1,06	0,25
Рівень захищеності даних	%	89	97	1,09	0,3
Дружність інтерфейсу	бал	4	9	2,25	0,15
Співвідношення ресурс/продуктивність	бал	5	8	1,6	0,2

Узагальнений коефіцієнт якості (Вн) для нового технічного рішення складе:

$$B_н = \sum_{i=1}^{k} \alpha_i \cdot \beta_i = 1{,}06 \cdot 0{,}1 + 1{,}06 \cdot 0{,}25 + 1{,}09 \cdot 0{,}3 + 2{,}25 \cdot 0{,}15 + 1{,}6 \cdot 0{,}2 = 1{,}36.$$

Отже за технічними параметрами, згідно узагальненого коефіцієнту якості розробки, науково-технічна розробка переважає існуючі аналоги приблизно в 1,36 рази.

4.3 Розрахунок витрат на проведення науково-дослідної роботи

Витрати, пов'язані з проведенням науково-дослідної роботи на тему «Інформаційна технологія тестування на проникнення мобільного застосунку», під час планування, обліку і калькулювання собівартості науково- дослідної роботи групуємо за відповідними статтями.

4.3.1 Витрати на оплату праці

До статті «Витрати на оплату праці» належать витрати на виплату основної та додаткової заробітної плати керівникам відділів, лабораторій, секторів і груп, науковим, інженерно-технічним працівникам, конструкторам, технологам, креслярам, копіювальникам, лаборантам, робітникам, студентам, аспірантам та іншим працівникам, безпосередньо зайнятим виконанням конкретної теми, обчисленої за посадовими окладами, відрядними розцінками, тарифнимиставками згідно з чинними в організаціях системами оплати праці.

Основна заробітна плата дослідників

Витрати на основну заробітну плату дослідників ($З_о$) розраховуємо у відповідності до посадових окладів працівників, за формулою [53]:

$$З_о = \sum_{i=1}^{k} \frac{M_{ni} \cdot t_i}{T_p},$$

де k – кількість посад дослідників залучених до процесу досліджень; M_{ni} – місячний посадовий оклад конкретного дослідника, грн; t_i – число днів роботи конкретного дослідника, дн.; T_p – середнє число робочих днів в місяці, T_p=21 дні.

$$З_о = 16950{,}00 \cdot 42 / 21 = 33900{,}00 \text{ грн.}$$

Проведені розрахунки зведемо до таблиці.

Таблиця 4.5 – Витрати на заробітну плату дослідників

Найменування посади	Місячний посадовийоклад, грн	Оплата за робочий день, грн	Число днів роботи	Витрати на заробітну плату, грн
Керівник проекту	16950,00	807,14	42	33900,00
Інженер-розробник програмного забезпечення	16750,00	797,62	38	30309,52
Консультант (фахівець напряму "Кібербезпека")	17000,00	809,52	5	4047,62
Лаборант	6800,00	323,81	15	4857,14
Всього				73114.28

Фінальне $З_о$ = 73114.28

Основна заробітна плата робітників

Витрати на основну заробітну плату робітників ($З_р$) за відповідними найменуваннями робіт НДР на тему «Інформаційна технологія тестування на проникнення мобільного застосунку» розраховуємо за формулою:

де $С_і$ – погодинна тарифна ставка робітника відповідного розряду, за виконану відповідну роботу, грн/год;

$t_і$ – час роботи робітника при виконанні визначеної роботи, год.

Погодинну тарифну ставку робітника відповідного розряду $С_і$ можна визначити за формулою:

$$С_і = \frac{М_М \cdot К_і \cdot К_с}{Т_р \cdot t_{зм}},$$

де $М_М$ – розмір прожиткового мінімуму працездатної особи, або мінімальної місячної заробітної плати (в залежності від діючого законодавства),приймемо $Мм$=6700,00 грн;

$К_і$ – коефіцієнт міжкваліфікаційного співвідношення для встановлення тарифної ставки робітнику відповідного розряду.

$К_с$ – мінімальний коефіцієнт співвідношень місячних тарифних ставок робітників першого розряду з нормальними умовами праці виробничих

об’єднань і підприємств до законодавчо встановленого розміру мінімальної заробітної плати.

$T_р$ – середнє число робочих днів в місяці, приблизно $T_р$ = 21 дн;

$t_{зм}$ – тривалість зміни, год.

C_1 = 6700,00 · 1,10 · 1,65 / (21 · 8) = 72,38 *грн.*

$З_{р1}$ = 72,38 · 8,00 = 579,07 *грн.*

Таблиця 4.7 – Величина витрат на основну заробітну плату робітників

Найменування робіт	Тривалість роботи, год	Тарифний коефіцієнт	Погодинна тарифна ставка, грн	Погодинна тарифна ставка, грн	Величина оплати на робітника
Установка обладнання для розробки програмного забезпечення	8,00	2	1,10	72,38	579,07
Інсталяція програмного забезпечення середовища розробки	6,20	3	1,35	88,83	550,78
Формування кодів програмних блоків	7,50	5	1,70	111,87	839,00
Формування бази даних для забезпечення машинного навчання	16,00	2	1,10	72,38	1158,14
Контроль проходження експериментів	8,00	4	1,50	98,71	789,64
Всього					3916.63

Фінальне Зр = 3916.63 грн

Додаткова заробітна плата дослідників та робітників

Додаткову заробітну плату розраховуємо як 10 … 12% від суми основної заробітної плати дослідників та робітників за формулою:

$$З_{дод} = (З_{о} + З_{р}) \cdot \frac{Н_{дод}}{100\%},$$

де $Н_{дод}$ – норма нарахування додаткової заробітної плати. Приймемо 11%.

$З_{дод}$ = (73114.29 + 3916.63) · 11 / 100% = 847340.12 грн.

4.3.2 Відрахування на соціальні заходи

Нарахування на заробітну плату дослідників та робітників розраховуємо як 22% від суми основної та додаткової заробітної плати дослідників і робітників за формулою:

$$З_н = (З_о + З_р + З_{дод}) \cdot \frac{Н_{зп}}{100\%}$$

де $Н_{зп}$ – норма нарахування на заробітну плату. Приймаємо 22%.

Зн = (73114.29 + 3916.63+847340.12) * 22 / 100% = 20336162.88 грн.

4.3.3 Сировина та матеріали

До статті «Сировина та матеріали» належать витрати на сировину, основні та допоміжні матеріали, інструменти, пристрої та інші засоби і предмети праці, які придбані у сторонніх підприємств, установ і організацій та витрачені на проведення досліджень за темою «Інформаційна технологія тестування на проникнення мобільного застосунку». Витрати на матеріали (М), у вартісному вираженні розраховуються окремопо кожному виду матеріалів за формулою:

$$M = \sum_{j=1}^{n} H_j \cdot Ц_j \cdot K_j - \sum_{j=1}^{n} B_j \cdot Ц_{вj},$$

де $Н_j$ – норма витрат матеріалу j-го найменування, кг; n – кількість видів матеріалів; $Ц_j$ – вартість матеріалу j-го найменування, грн/кг; $К_j$ – коефіцієнт транспортних витрат, ($К_j$ = 1,1 … 1,15);$В_j$ – маса відходів j-го найменування, кг; $Ц_{вj}$ – вартість відходів j-го найменування, грн/кг.

М = 3,0 · 225,00 · 1,11 - 0 · 0 = 749,25 грн.

Таблиця 4.7 – Витрати на матеріали

Найменування матеріалу, марка, тип, сорт	Ціна за 1 кг, грн	Норма витрат, кг	Величина відходів, кг	Ціна відходів, грн/кг	Вартість витраченого матеріалу, грн
Папір офісний Mondi A4 80 г/м	225,00	3,0	0	0	749,25
Набір настільний 7015 - 08 SCHOLZ	520,00	2,0	0	0	391.00
Всього					1140.25

Фінальне М = 1140.25

4.3.4 Розрахунок витрат на комплектуючі

Витрати на комплектуючі *($К_в$)*, які використовують при проведенні НДР на тему «Інформаційна технологія тестування на проникнення мобільного застосунку», розраховуємо, згідно з їхньою номенклатурою, за формулою:

$$К_в = \sum_{j=1}^{n} Н_j \cdot Ц_j \cdot К_j$$

де $Н_j$ – кількість комплектуючих j-го виду, шт.; $Ц_j$ – покупна ціна комплектуючих j-го виду, грн;

$К_j$ – коефіцієнт транспортних витрат, ($К_j$ = *1,1 … 1,15*).

$К_в = 2 \cdot 869.36 \cdot 1,11 = 1929.9792$грн.(округлимо до 1930)

Таблиця 4.8 – Витрати на комплектуючі

Найменування комплектуючих	Кількість, шт.	Ціна за штуку, грн	Сума, грн
Оперативна пам'ять:12GB DDR5	1	869.36	1930
Маршрутизатор TP-Link ARCHER-C80	1	1799	2000
Всього			3930

Фінальне $К_в$ = 3930 грн

4.1.5 Спецустаткування для наукових (експериментальних) робіт

До статті «Спецустаткування для наукових (експериментальних) робіт» належать витрати на виготовлення та придбання спецустаткування необхідного для проведення досліджень, також витрати на їх проектування, виготовлення, транспортування, монтаж та встановлення.

Балансову вартість спецустаткування розраховуємо за формулою:

$$В_{спец} = \sum_{i=1}^{k} Ц_i \cdot С_{пр.i} \cdot К_i ,$$

де Ці – ціна придбання одиниці спецустаткування даного виду, марки, грн;

$С_{пр.і}$ –кількість одиниць устаткування відповідного найменування, які придбані для проведення досліджень, шт.;

$К_i$ – коефіцієнт, що враховує доставку, монтаж, налагодження устаткуваннятощо, (*$К_i$ = 1,10…1,12);*

k – кількість найменувань устаткування.

$В_{спец}$ = 8999,00 · 1 · 1,1 = 9898,90 грн.

Витрата на придбання спецустаткування це телефон Xiaomi mi8 8/128 gb в кількості 1 шт. Фінальне $В_{спе}$ = 9898,90

4.3.5 Програмне забезпечення для наукових (експериментальних) робіт

До статті «Програмне забезпечення для наукових (експериментальних) робіт» належать витрати на розробку та придбання спеціальних програмних засобів і програмного забезпечення, (програм, алгоритмів, баз даних) необхідних для проведення досліджень, також витрати на їх проектування, формування та встановлення.

Балансову вартість програмного забезпечення розраховуємо за формулою:

$$B_{прг} = \sum_{i=1}^{k} Ц_{іпрг} \cdot C_{прг.і} \cdot K_i ,$$

де Ціпрг – ціна придбання одиниці програмного засобу даного виду, грн;

Спрг.і – кількість одиниць програмного забезпечення відповідного найменування, які придбані для проведення досліджень, шт.;

Кі – коефіцієнт, що враховує інсталяцію, налагодження програмного засобутощо, (*Кі = 1,10…1,12); k* – кількість найменувань програмних засобів.

Для програмного ПЗ використовувалося Android Studio яке є безкоштовним, тому фінальне $B_{прг} = 0$ *грн*

4.3.6 Амортизація обладнання, програмних засобів та приміщень

В спрощеному вигляді амортизаційні відрахування по кожному виду обладнання, приміщень та програмному забезпеченню тощо, розраховуємо з використанням прямолінійного методу амортизації за формулою:

$$A_{обл} = \frac{Ц_б}{T_в} \cdot \frac{t_{вик}}{12},$$

де Цб – балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

tвик – термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

Тв – строк корисного використання обладнання, програмних засобів, приміщень тощо, років.

$A_{обл} = (56899{,}00 \cdot 2) / (2 \cdot 12) = 4741{,}58$ *грн.*

Проведені розрахунки зведемо до таблиці.

Таблиця 4.11 – Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн	Строк корисного використання, років	Термін використання обладнання, місяців	Амортизаційні відрахування, грн
Ноутбук ASUS ROG Strix G16 G614JI / i9-13980HX / RTX 4070	56899,00	2	2	4741,58
Робоче місце інженера-розробника ПЗ	8700,00	5	2	290,00
Пристрої передачі даних комутатор мережевий TP- Link	2780,00	2	1	231,67
Пристрій виводу інформації	6740,00	5	5	224,67
Приміщення лабораторії	725000,00	25	25	4833,33
Всього				10321.25

Фінальне $A_{обл}$ = 10321.25

4.3.7 Паливо та енергія для науково-виробничих цілей

Витрати на силову електроенергію (Ве) розраховуємо за формулою:

$$B_e = \sum_{i=1}^{n} \frac{W_{yi} \cdot t_i \cdot Ц_e \cdot K_{впі}}{\eta_i},$$

де W_{yi} – встановлена потужність обладнання на визначеному етапі розробки, кВт; ti – тривалість роботи обладнання на етапі дослідження, год; $Ц_e$ – вартість 1 кВт-години електроенергії, грн; (вартість електроенергії визначається за даними енергопостачальної компанії), приймемо Це = 6,20 грн; $К_{впі}$ – коефіцієнт, що враховує використання потужності, $К_{впі}$ <1; і – коефіцієнт корисної дії обладнання, і<1.

Ве = 0.04 · 310.0 · 6.20 · 0.95 / 0.97 = 76,88 грн.

Проведені розрахунки зведемо до таблиці.

Найменування обладнання	Встановлена потужність, кВт	Тривалість роботи, год	Сума, грн
Ноутбук ASUS ROG Strix G16 G614JI / i9-13980HX / RTX 4070	0.28	280	76,88

Фінальне Be = 76,88 грн

4.3.8 Службові відрядження

Витрати на службові відрядження на тему «Інформаційна технологія тестування на проникнення мобільного застосунку» відсутні.

4.3.9 Витрати на роботи, які виконують сторонні підприємства, установи і організації

Витрати на роботи, які виконують сторонні підприємства, установи і організації тему «Інформаційна технологія тестування на проникнення мобільного застосунку» відсутні.

4.3.10 Інші витрати

До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками. Для теми «Інформаційна технологія тестування на проникнення мобільного застосунку» такі витрати відсутні.

4.3.11 Накладні (загальновиробничі) витрати

До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов’язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов’язані з набором робочої сили; витрати на оплату послуг банків; витрати, пов’язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін.

Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуємо як 100…150% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}$$

де $Н_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати», приймемо $Н_{нзв}$ = 100%.

$В_{нзв}$ = (73114.28 + 3916.63) · 100 / 100% = 7703091 грн.

Витрати на проведення науково-дослідної роботи на тему «Інформаційна технологія тестування на проникнення мобільного застосунку» розраховуємо як суму всіх попередніх статей витрат за формулою:

$$В_{заг} = З_о + З_р + З_{дод} + З_н + М + К_в + В_{спец} + В_{прг} + А_{обл} + В_е + В_{св} + В_{сп} + І_в + В_{нзв}$$

$$В_{заг} = 73114.28 + 3916.63 + 847340.12 + 20336162.88 + 1140.25 + 3930 + 9898.90 + 0 +$$

$$10321.25 + 76.88 + 0 + 0 + 0 + 7703091 = 28988992.19 \text{ грн}$$

Загальні витрати ЗВ на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховується за формулою:

$$ЗВ = \frac{В_{заг}}{\eta}$$

де - коефіцієнт, який характеризує етап (стадію) виконання науководослідної роботи, приймемо =0,9.

$$ЗВ = 28988992.19 / 0.9 = 32209991.32 \text{ грн.}$$

4.4 Розрахунок економічної ефективності науково-технічної розробки

В ринкових умовах узагальнюючим позитивним результатом, що його може отримати потенційний інвестор від можливого впровадження результатів тієї чи іншої науково-технічної розробки, є збільшення у потенційного інвестора величини чистого прибутку.

Результати дослідження проведені за темою «Інформаційна технологія тестування на проникнення мобільного застосунку» передбачають комерціалізацію протягом 4-х років реалізації на ринку.

В цьому випадку основу майбутнього економічного ефекту будуть формувати:

N – збільшення кількості споживачів яким надається відповідна інформаційна послуга у періоди часу, що аналізуються;

Таблиця 4.13 – Збільшення кількості споживачів

Показник	1-й рік	2-й рік	3-й рік	4-й рік
Збільшення кількості споживачів, осіб	1000	2000	2500	2000

N – кількість споживачів яким надавалась відповідна інформаційна послуга у році до впровадження результатів нової науково-технічної розробки, приймемо 15000 осіб;

$Ц_б$ – вартість послуги у році до впровадження інформаційної системи, приймемо 8150,00 грн

– зміна вартості послуги від впровадження результатів, приймемо 397,32 грн.

Можливе збільшення чистого прибутку у потенційного інвестора $П_i$ для кожного із 4-х років, протягом яких очікується отримання позитивних результатів від можливого впровадження та комерціалізації науково-технічної розробки, розраховуємо за формулою:

$$\Delta П_i = (\pm \Delta Ц_o \cdot N + Ц_o \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\vartheta}{100}),$$

де λ – коефіцієнт, який враховує сплату потенційним інвестором податку на додану вартість. У 2023 році ставка податку на додану вартість складає 20%, а коефіцієнт λ =0,8333;

ρ – коефіцієнт, який враховує рентабельність інноваційного продукту).

Приймемо ρ =40%;

ϑ – ставка податку на прибуток, який має сплачувати потенційний інвестор, у 2023 році ϑ =18%;

Збільшення чистого прибутку 1-го року:

П1=(397,32·15000,00+8547,33·1000)·0,83·0,4·(1-0,18/100%)=3949440,13 грн.

Збільшення чистого прибутку 2-го року:

П2=(397,32·15000,00+8547,33·3000)·0,83·0,4·(1-0,18/100%)=8603287,64 грн.

Збільшення чистого прибутку 3-го року:

П3=(397,32·15000,00+8547,33·5500)·0,83·0,4·(1-0,18/100%)=14420597,04 грн.

Збільшення чистого прибутку 4-го року:

П4=(397,32·15000,00+8547,33·7500)·0,83·0,4·(1-0,18/100%)=19074444,56 грн.

Приведена вартість збільшення всіх чистих прибутків ПП, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки:

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t},$$

де Пі – збільшення чистого прибутку у кожному з років, протягом яких виявляються результати впровадження науково-технічної розробки, грн;

Т – період часу, протягом якого очікується отримання позитивних результатів від впровадження та комерціалізації науково-технічної розробки, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, τ =0,27;

t – період часу (в роках) від моменту початку впровадження науково-технічної розробки до моменту отримання потенційним інвестором додаткових чистих прибутків у цьому році.

ПП =3949440,13/(1+0,27)^1+8603287,64/(1+0,27)^2+14420597,04/(1+0,27)^3+
+19074444,56/(1+0,27)^4=3109795,38+5334049,01+7039990,59+7332245,82
=22816080,79 грн.

Величина початкових інвестицій PV , які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки:

$$PV = k_{інф} \cdot ЗВ,$$

Де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію, приймаємо $k_{інв}$=2,1;

ЗВ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, приймаємо 32209991.32 грн

PV = $k_{інв}$ * ЗВ= 2.1 * 32209991.32 = 67640981.772 грн.

Абсолютний економічний ефект $E_{абс}$ для потенційного інвестора від можливого впровадження та комерціалізації науково-технічно розробки становитиме:

$$E_{абс} = ПП - PV$$

де ПП – приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково-технічної розробки, 222816080.79грн;

PV – теперішня вартість початкових інвестицій, 67640981.772 грн.

$$E_{абс} = ПП - PV = 222816080.79 - 67640981.772 = 155175099.018 \text{ грн.}$$

Внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$E_в = \sqrt[Tж]{1 + \frac{E_{абс}}{PV}} - 1,$$

де Еабс – абсолютний економічний ефект вкладених інвестицій, *155013496.038* грн;

PV – теперішня вартість початкових інвестицій, 67640981.772 грн;

$T_ж$ – життєвий цикл науково-технічної розробки, тобто час від початку її

Розробки до закінчення отримування позитивних результатів від її впровадження, 4 роки.

$$Ев = (1+155175099.018 /67640981.772) = 3.29$$

Мінімальна внутрішня економічна дохідність вкладених інвестицій мін:

$$\tau_{мін} = d + f,$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2023 році в Україні d =0,1;

f – показник, що характеризує ризикованість вкладення інвестицій, приймемо 0,27.

$\tau_{мін}$ = 0,1+0,27 = 0,37 < 3.29 свідчить про те, що внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки вища мінімальної внутрішньої дохідності. Тобто інвестувати в науково-дослідну роботу за темою «Інформаційна технологія тестування на проникнення мобільного застосунку» доцільно.

Період окупності інвестицій $T_{ок}$ які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$T_{ок} = \frac{1}{E_в},$$

де $E_в$ – внутрішня економічна дохідність вкладених інвестицій.

$T_{ок}$ = 1 / 3.29 = 0.3039 р.

$T_{ок}$ < 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок.

Висновки до розділу

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Інформаційна технологія тестування на проникнення мобільного застосунку» становить 39,3 бала, що, свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки вище середнього).

ВИСНОВКИ

Під час виконання магістерської кваліфікаційної роботи було проаналізовано завдання, визначені завдання та поставлені вимоги, дотримавшись яких, можна досягти поставленої мети. Було проаналізовано актуальність вирішення задачі тестування на проникнення мобільного застосунку у кібербезпеці.

Проведено аналіз методів та існуючих засобів тестування на проникнення мобільного застосунку. Визначено, що проблема є доволі актуальною. Прийнято рішення покращити процес тестування на проникнення мобільного застосунку шляхом розробки інформаційної технології.

Були проаналізовані методи та засоби тестування мобільних застосунків на проникнення. Після аналізу та порівняння методів та засобів тестування мобільного застосунку на проникнення було визначено, що OWASP є найкращим методом тестуванням мобільного застосунку на проникнення, але мінусом кожного із методів є те що вони не надають інформація як створювати звіт з тестування та який метод да засіб потрібно використовувати для тестування різних типів застосунків.

Проведено тестування розробленої інформаційної технології. Після тестування було виявлено ряд помилок, до яких йшов опис про серйозність ризику виникнення загрози, ймовірність виникнення загрози, вплив на застосунок та було надано рекомендації що потрібно зробити, щоб уникнути тої чи іншої загрози безпеки застосунку та даних користувача.

Дану інформаційну технологію можна використовувати тестування на проникнення мобільного застосунку на проникнення. Інформаційна технологія складається з різних етапів, тому її легко адаптувати під потрібні задачі, розширити або змінити функціонал.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Mobile App Security: A Comprehensive Guide to Penetration Testing: веб-сайт. URL: https://qualysec.com/mobile-application-penetration-testing-a-guide/ (дата звернення 20.09.2023)
2. Android Platform architecture: веб-сайт. URL: https://developer.android.com/guide/platform (дата звернення 22.09.2023)
3. Android Operating System: веб-сайт. URL: https://www.javatpoint.com/android-operating-system (дата звернення 23.09.2023)
4. Android <permission>: веб-сайт. URL: https://developer.android.com/guide/topics/manifest/permission-element (дата звернення 26.09.2023)
5. What are The Different Protection Levels in Android Permission? >: веб-сайт. URL: https://www.geeksforgeeks.org/what-are-the-different-protection-levels-in-android-permission/ (дата звернення 29.09.2023)
6. Permissions on Android: веб-сайт. URL: https https://developer.android.com/guide/topics/permissions/overview (дата звернення 29.09.2023)
7. Smartphone Security and Privacy: A Survey on APTs, Sensor-Based Attacks, Side-Channel Attacks, Google Play Attacks, and Defenses: веб-сайт. URL: https://www.mdpi.com/2227-7080/11/3/76 (дата звернення 06.10.2023)
8. About Android App Bundles: веб-сайт. URL: https://developer.android.com/guide/app-bundle (дата звернення 10.10.2023)
9. How Google Play works: веб-сайт. URL: https://play.google/howplayworks/ (дата звернення 10.10.2023)
10. Mobile App Security: A Comprehensive Guide to Penetration Testing: веб-сайт. URL: https://qualysec.com/mobile-application-penetration-testing-a-guide/ (дата звернення 16.10.2023)
11. OWASP Mobile Application Security Mobile Application Security Testing - Penetration Testing (a.k.a. Pentesting) : веб-сайт. URL: https://mas.owasp.org/MASTG/General/0x04b-Mobile-App-Security-

Testing/#penetration-testing-aka-pentesting (дата звернення 18.10.2023)

12. OWASP Mobile Application Security: веб-сайт. URL: https://mas.owasp.org/ (дата звернення 18.10.2023)

13. NIST SP 800-163 Rev. 1 - Vetting the Security of Mobile Applications: веб-сайт. URL: https://csrc.nist.gov/pubs/sp/800/163/r1/final (дата звернення 18.10.2023)

14. Mobile Application Security Testing (MAST) - Challenges & Tools: веб-сайт. URL: https://snyk.io/learn/application-security/mobile-application-security/mast-mobile-app-sec-testing/ (дата звернення 18.10.2023)

15. Vetting the Security of Mobile Applications: NIST Publishes SP 800-163 Revision 1: веб-сайт. URL: https://csrc.nist.gov/news/2019/nist-publishes-sp-800-163-rev-1 (дата звернення 18.10.2023)

16. NIST SP 800-163 Rev. 1 (Initial Public Draft): веб-сайт. URL: https://www.iso.org/standard/75652.html (дата звернення 18.10.2023)

17. PCI DSS (Payment Card Industry Data Security Standard): веб-сайт. URL: https://csrc.nist.gov/pubs/sp/800/163/r1/ipd (дата звернення 18.10.2023)

18. Sonarhttps: веб-сайт. URL: www.sonarsource.com/ (дата звернення 26.10.2023)

19. Checkmarx: веб-сайт. URL: https://checkmarx.com/ (дата звернення 26.10.2023)

20. Fortify: веб-сайт. URL: https://www.microfocus.com/en-us/cyberres/application-security (дата звернення 26.10.2023)

21. config-check: веб-сайт. URL: https://www.npmjs.com/package/config-check (дата звернення 26.10.2023)

22. Kali - The most advanced Penetration Testing Distribution: веб-сайт. URL: https://www.kali.org/ (дата звернення 05.11.2023)

23. Android Studio: веб-сайт. URL: https://developer.android.com/studio (дата звернення 05.11.2023)

24. IntelliJ IDEA – the Leading Java and Kotlin IDE: веб-сайт. URL: https://www.jetbrains.com/idea (дата звернення 11.11.2023)

25. Burp Suite - Application Security Testing Software: веб-сайт. URL: https://portswigger.net/burp (дата звернення 11.11.2023)

26. Wireshark: веб-сайт. URL: https://www.wireshark.org/ (дата звернення 11.11.2023)

27. Frida. Dynamic instrumentation toolkit for developers, reverse-engineers, and security researchers.: веб-сайт. URL: https://frida.re/ (дата звернення 11.11.2023)

28. Drozer. Comprehensive security and attack framework for Android.: веб-сайт. URL: https://labs.withsecure.com/tools/drozer (дата звернення 11.11.2023)

29. EncryptedSharedPreferences.: веб-сайт. URL: https://developer.android .com/reference/androidx/security/crypto/EncryptedSharedPreferences (дата звернення 12.11.2023)

30. Android Keystore system.: веб-сайт. URL: https://developer .android.com/privacy-and-security/keystore (дата звернення 12.11.2023)

31. Glide.: веб-сайт. URL: https://github.com/bumptech/glide (дата звернення 12.11.2023)

32. Sensitive Data - OWASP Mobile Application Security.: веб-сайт. URL: https://mas.owasp.org/MASTG/tests/android/MASVS-STORAGE/MASTG-TEST-0003/ (дата звернення 23.11.2023)

33. Testing Obfuscation - OWASP Mobile Application Security.: веб-сайт. URL: https://mas.owasp.org/MASTG/tests/android/MASVS-RESILIENCE/MASTG-TEST-0051/ (дата звернення 23.11.2023)

34. Finding Sensitive Information in Auto-Generated Screenshots - OWASP Mobile Application Security: веб-сайт. URL: https://mas.owasp.org/MASTG/ tests/android/MASVS-PLATFORM/MASTG-TEST-0010/#static-analysis

35. Testing Root Detection - OWASP Mobile Application Security: веб-сайт. URL: https://mas.owasp.org/MASTG/tests/android/MASVS-RESILIENCE/MASTG

TEST-0045/ (дата звернення 23.11.2023)

36. Testing Emulator Detection - OWASP Mobile Application Security: веб-сайт. URL: https://mas.owasp.org/MASTG/tests/android/MASVS-RESILIENCE/MASTG-TEST-0049/ (дата звернення 23.11.2023)

37. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт / Уклад. : В. О. Козловський, О. Й. Лесько, В. В. Кавецький. – Вінниця : ВНТУ, 2021. – 42 с.

38. Кавецький В. В. Економічне обґрунтування інноваційних рішень: практикум / В. В. Кавецький, В. О. Козловський, І. В. Причепа – Вінниця : ВНТУ, 2016. – 113 с.

39. Mobile Security Framework (MobSF) : веб-сайт. URL: https://github.com/MobSF/Mobile-Security-Framework-MobSF (дата звернення 07.12.2023)

40. AppUse PRO - Let AppUse Pro do the work for you! : веб-сайт. URL: https://appsec-labs.com/appuse/ (дата звернення 07.12.2023)

41. Quick Android Review Kit: веб-сайт. URL: github.com/linkedin/qark (дата звернення 07.12.2023)

42. AndroBugs Framework: веб-сайт. URL: https://github.com/AndroBugs/AndroBugs_Framework (дата звернення 07.12.2023)

43. Xposed Framework: What It Is and How to Install It: веб-сайт. URL: lifewire.com/xposed-framework-4148451 (дата звернення 07.12.2023)

44. Burp Suite Mobile Assistant: веб-сайт. URL: yw9381.github.io/Burp_Suite_Doc_en_us/burp/documentation/desktop/tools/mobile-assistant/index.html (дата звернення 07.12.2023)

ДОДАТКИ

Додаток А

ПРОТОКОЛ ПЕРЕВІРКИ МАГІСТЕРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: Інформаційна технологія тестування на проникнення мобільного застосунку

Автор роботи: Ворожбит Михайло Вікторович

Тип роботи: магістерська кваліфікаційна робота

Підрозділ кафедра захисту інформації ФІТК
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність – 96,5 %. Схожість – 3,5 %.

Аналіз звіту подібності (відмітити потрібне):

1. Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

2. Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостій-ності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

3. Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку _____ Валентина КАПЛУН
(підпис)

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи _____ Михайло ВОРОЖБИТ
(підпис)

Керівник роботи _____ Леонід КУПЕРШТЕЙН
(підпис)

Додаток А

ПРОТОКОЛ ПЕРЕВІРКИ МАГІСТЕРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: Інформаційна технологія тестування на проникнення мобільного застосунку

Автор роботи: Ворожбит Михайло Вікторович

Тип роботи: магістерська кваліфікаційна робота

Підрозділ: кафедра захисту інформації ФІТК
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність – 96,5 %. Схожість – 3,5 %.

Аналіз звіту подібності (відмітити потрібне):

1. Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

2. Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостій-ності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

3. Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку _____ (підпис) Валентина КАПЛУН

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи _____ (підпис) Михайло ВОРОЖБИТ

Керівник роботи _____ (підпис) Леонід КУПЕРШТЕЙН

Додаток Б

Звіт про випробування на проникнення Android застосунку

Зміст:

1.0 Методологія розрахунку ризиків

Фактори ризику «Risk Factors»: Кожному результату присвоюється два фактори для вимірювання його ризику. Фактори вимірюються за шкалою від 1 (low), 2 (medium) до 3 (high).

Вплив «Impact»: Вказує на вплив виявленого фактору на технічні та бізнес-операції. Він охоплює такі аспекти, як як конфіденційність, цілісність і доступність даних або систем, а також фінансові або фінансові або репутаційні втрати.

Ймовірність «Likelihood»: Вказує на потенційну можливість використання результатів дослідження. Вона враховує такі аспекти, як рівень кваліфікації зловмисника рівень кваліфікації зловмисника та відносна легкість використання.

Високий «High»: Вразливості з високим або більшим впливом на бізнес і високою або більшою ймовірністю вважаються вразливими з високим ступенем серйозності. Оцінка ризику мінімум 6.

Середній «Medium»: Вразливості з помірним впливом на бізнес та ймовірністю вважаються серйозними "Medium". Сюди також відносяться

вразливості, які мають або дуже високий вплив на бізнес у поєднанні з низькою ймовірністю, або мають низький вплив на бізнес у поєднанні з дуже високою ймовірністю. ймовірність. Оцінка ризику від 3 до 6.

Низький «Low»: Вразливості, які мають або дуже низький вплив на бізнес, або максимально високу ймовірність, або дуже дуже низьку ймовірність, але максимально високий вплив на бізнес, вважаються вразливостями з низьким ступенем серйозності. Крім того, вразливості, для яких і вплив, і ймовірність є низькими, вважаються низькими серйозність. Оцінка ризику від 0 до 3.

2.0 Короткий зміст

Проведено тест на проникнення в андроїд-Застосунок PROD_APP.

ІНФОРМАЦІЯ ПРО ЗАСТОСУНОК

Назва програми: PROD_APP

Назва пакету: prod

Основна діяльність: SplashActivity

Цільовий SDK: 33

Min SDK: 23

Назва версії для Android: 1.0.0.0.0-prod

Код версії Android: 100000

APK підписано

v1 signature: True

v2 signature: True

v3 signature: False

Ця оцінка "білої скриньки" та "чорної скрьки" була проведена для виявлення лазівок у застосунку з точки зору з точки зору безпеки. Ця оцінка була спрямована на виявлення вразливостей, присутніх в застосунку, які можуть призвести до ін'єкції, витоку інформації та інших ризиків, що можуть спричинити потенційних бізнес-збитків.

У цьому звіті представлені результати оцінки.

Загалом, під час оцінювання було виявлено декілька результатів, які будуть детально описані буде надано в розділі "Висновки".

2.1 Конфіденційні дані в журналах (Sensitive data in logs)

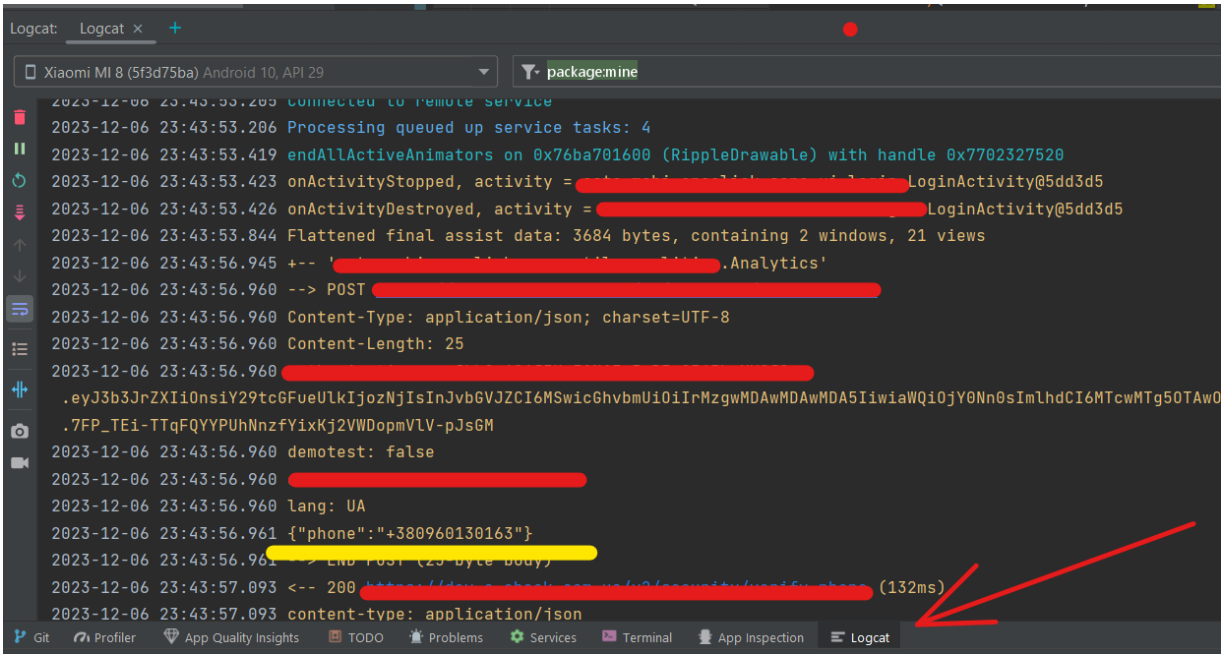
Ступінь ризику: Високий

Ймовірність: Середній

Вплив: Високий

Вектор атаки: Індивідуальний користувач

Опис: Застосунок використовує запис для реєстрації конфіденційну інформацію, таку як токен JWT користувача, дані користувача, ім'я та прізвище користувача.



Наслідки: У разі отримання зловмисником доступу до телефону (фізичного або віддаленого) він може встановити збір логів.

Рекомендація:

- Вимкніть збір конфіденційних даних.

Посилання: Sensitive Data - OWASP Mobile Application Security [32].

2.2 Відсутність належного обфускування коду застосунку

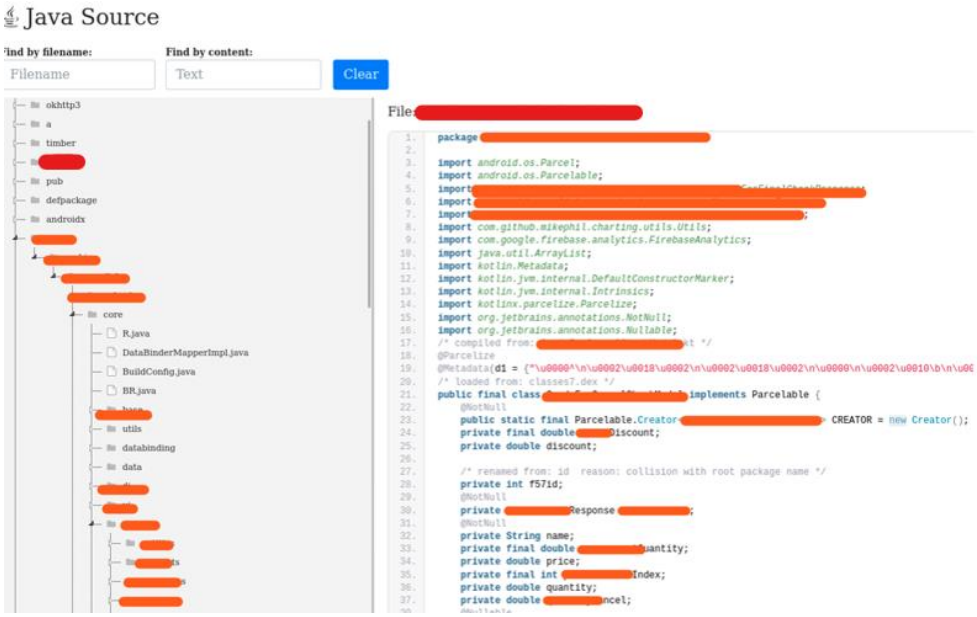
Серйозність ризику: Середній

Ймовірність: Низька

Вплив: Середній

Вектор атаки: Індивідуальний користувач

Опис: Застосунок було скомпільовано без належного обфускації коду. У декомпільованому програма показує всі імена класів та параметрів. Обфускація коду може значно збільшити складність розуміння логіки роботи програми.



Вплив: Зловмисник може легко декомпілювати код, зрозуміти логіку програми та використати ці знання для подальших атак.

Рекомендація:

- Потрібно робити обфускацію коду перед компіляцією.

Посилання: Sensitive Data - OWASP Mobile Application Security [33].

2.3 Автоматично згенеровані скріншоти дій з конференційними даними

Серйозність ризику: Низька

Ймовірність: Низька

Вплив: Низький

Вектор атаки: Індивідуальний користувач

Опис: Знімок екрана поточної активності робиться, коли програма Android переходить у фоновий режим, і відображається з естетичною метою, коли програма повертається на передній план. Однак це може призвести до витоку конфіденційної інформації.

Вплив: Зловмисник може отримати доступ до папки з автоматично згенерованими знімками екрана, що може призвести до розкриття інформації.

Рекомендація:

- Встановіть прапорець "БЕЗПЕЧНО" для дій з конфіденційною інформацією

Довідка: Finding Sensitive Information in Auto-Generated Screenshots - OWASP Mobile Application Security [34].

2.4 Відсутність конфігурації відмовостійкості

Серйозність ризику: Низька

Ймовірність: Низька

Вплив: Низький

Вектор атаки: Індивідуальний користувач

Опис: Програма не виявляє рутовані, емульовані або незашифровані пристрої.

Вплив: Зловмисник може використати цю ваду для тестування програми, щоб знайти слабкі місця або маніпулювати нею.

Рекомендація:

- Налаштувати виявлення кореневих, емульованих, незашифрованих пристроїв

Посилання: Testing Root Detection - OWASP Mobile Application Security [35], Testing Emulator Detection - OWASP Mobile Application Security [35].

3.0 Висновки

Під час проведення тесту на проникнення в андроїд застосунок було виявлено кілька критичних вразливостей, які можуть стати причиною серйозних ризиків для безпеки застосунку та конфіденційної інформації користувача.

Додаток В

ЗАТВЕРДЖУЮ
Директор ТОВ «ASTA.MOBI»
_____ Андрій СТАХОВ
«___» _____ 2023 р.

АКТ
про впровадження результатів магістерської кваліфікаційної роботи
Ворожбита Михайла Вікторовича

Комісія у складі: голова комісії – директор Андрій Стахов, розглянувши матеріали магістерської роботи Михайла Ворожбита на тему «Інформаційна технологія тестування на проникнення мобільного застосунку», склала цей акт про те, що у ТОВ «ASTA.MOBI» впроваджено результати цієї роботи, а саме результат тестування мобільного застосунку на проникнення.

Впроваджені результати дозволяють покращити безпеку мобільних застосунків.

Голова комісії:
Директор _____ Андрій СТАХОВ

ЗАТВЕРДЖУЮ
Директор ТОВ «ASTA.MOBI»
_____ Андрій СТАХОВ
«___» _____ 2023 р.

АКТ
про впровадження результатів магістерської кваліфікаційної роботи
Ворожбита Михайла Вікторовича

Комісія у складі: голова комісії – директор Андрій Стахов, розглянувши матеріали магістерської роботи Михайла Ворожбита на тему «Інформаційна технологія тестування на проникнення мобільного застосунку», склала цей акт про те, що у ТОВ «ASTA.MOBI» впроваджено результати цієї роботи, а саме результат тестування мобільного застосунку на проникнення.

Впроваджені результати дозволяють покращити безпеку мобільних застосунків.

Голова комісії:
Директор

Товариство з обмеженою відповідальністю "АСТА.МОБІ" Ідент.код 41051004 * УКРАЇНА *

Андрій СТАХОВ

Додаток Г

ІЛЮСТРАТИВНА ЧАСТИНА

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ ТЕСТУВАННЯ НА ПРОНИКНЕННЯ МОБІЛЬНОГО ЗАСТОСУНКУ

СХЕМА ПРОЦЕСУ ТЕСТУВАННЯ НА ПРОНИКНЕННЯ МОБІЛЬНИХ ЗАСТОСУНКІВ

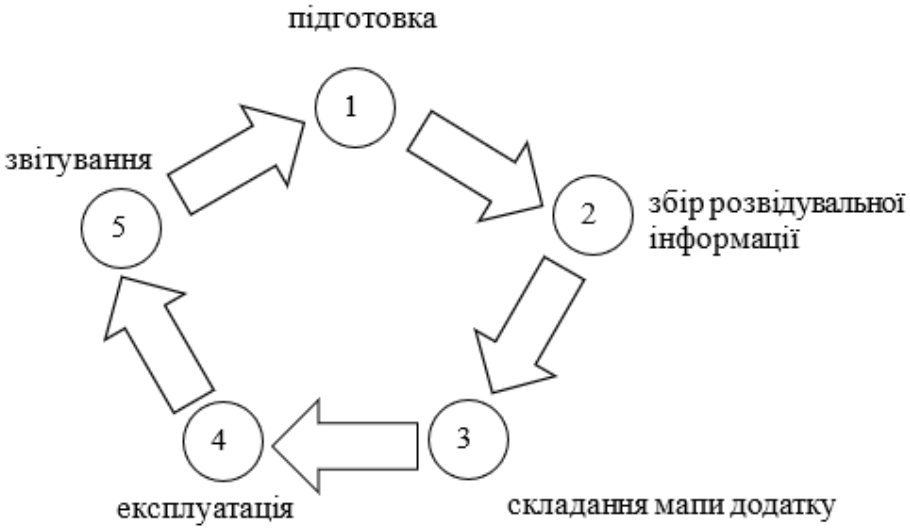


СХЕМА ПРОЦЕСІВ ТЕХНОЛОГІЇ НА ТЕСТУВАННЯ ПРОНИКНЕННЯ

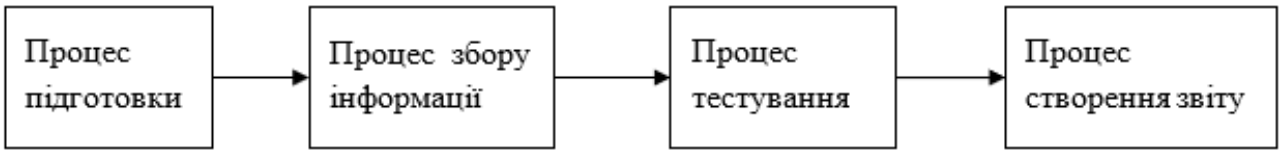


СХЕМА ПРОЦЕСУ ПІДГОТОВКИ

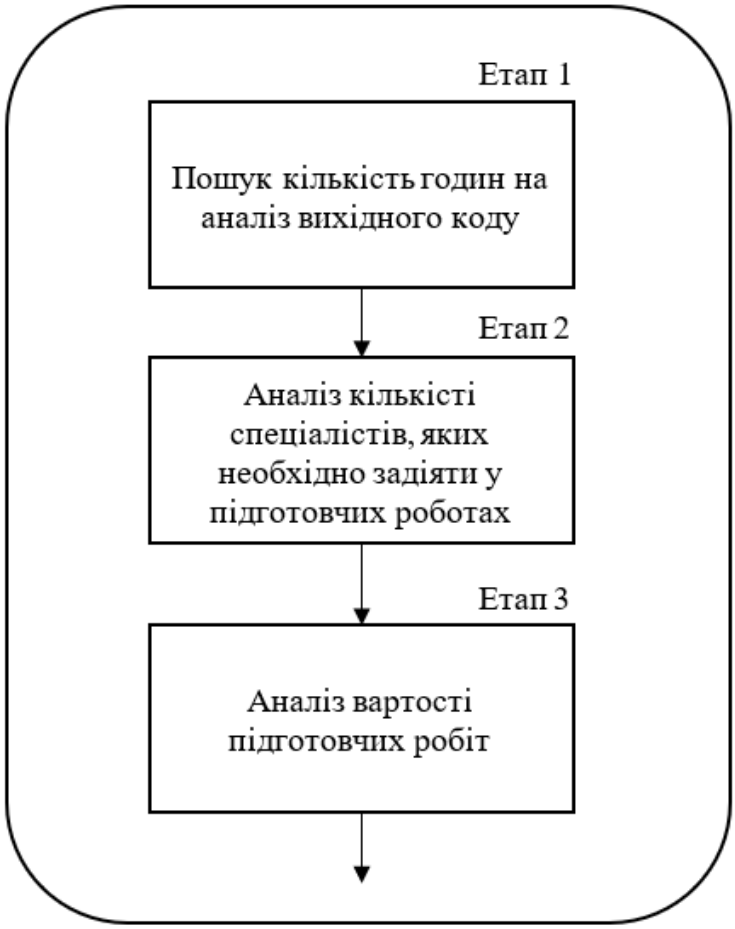


СХЕМА ПРОЦЕСУ ЗБОРУ ІНФОРМАЦІЇ

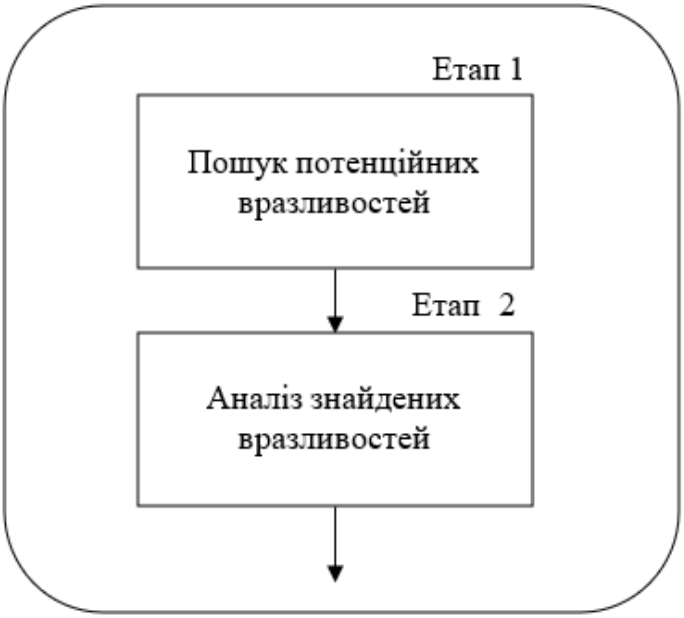
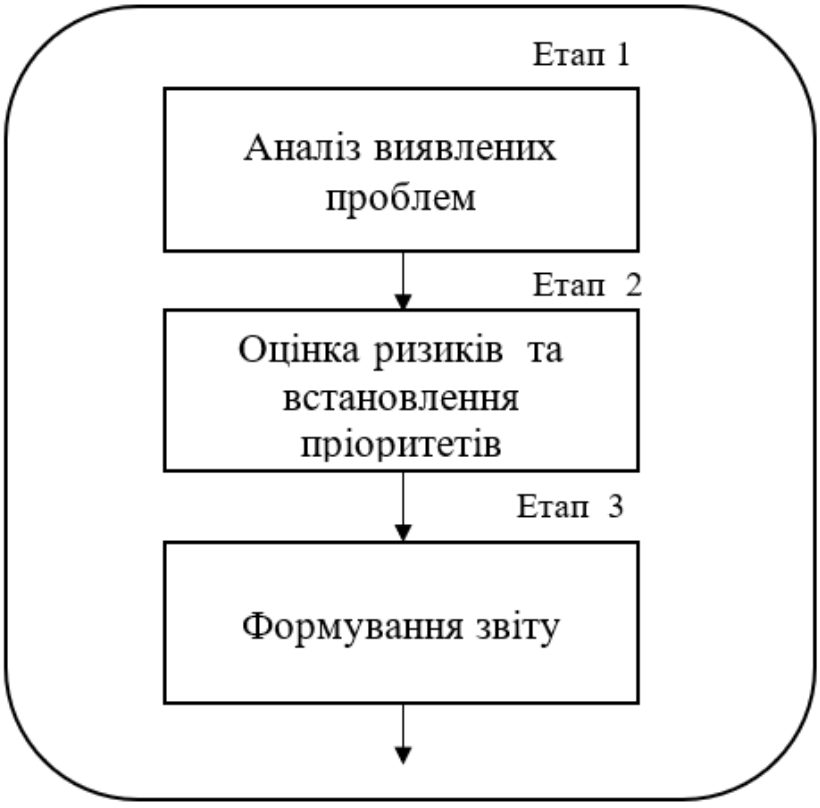


СХЕМА ПРОЦЕСУ ТЕСТУВАННЯ

СХЕМА ПРОЦЕСУ СТВОРЕННЯ ЗВІТУ



ЗАГАЛЬНА СХЕМА ПРОЦЕСУ ТЕСТУВАННЯ МОБІЛЬНОГО ЗАСТОСУНКУ НА ПРОНИКНЕННЯ

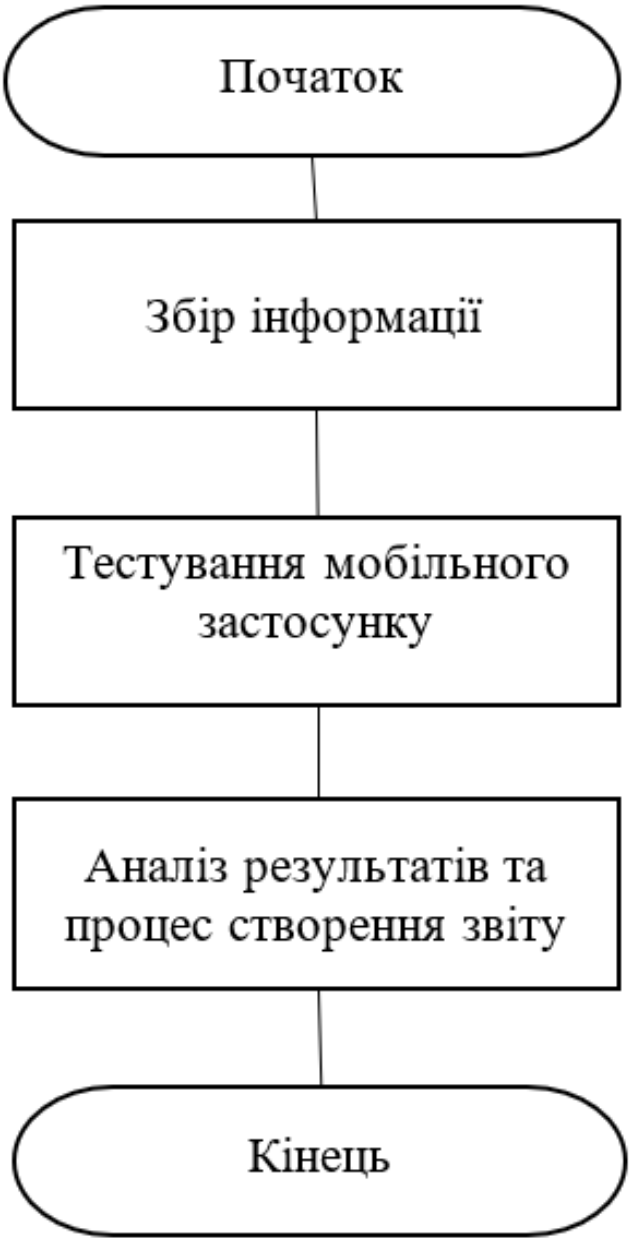


СХЕМА АЛГОРИТМУ ЗБОРУ ІНФОРМАЦІЇ

СХЕМА АЛГОРИТМУ ТЕСТУВАННЯ МОБІЛЬНОГО ЗАСТОСУНКУ

СХЕМА АЛГОРИТМУ СТВОРЕННЯ ЗВІТУ