COMPUTER SYSTEMS AND NETWORKS

Vol. 7, No. 1, 2025

DOI: https://doi.org/10.23939/csn2025.01.284

УДК 621.3

АНАЛІЗ БЕЗПЕКОВИХ РИЗИКІВ У МОБІЛЬНИХ ДОДАТКАХ, СТВОРЕНИХ НА КРОСПЛАТФОРМНИХ ФРЕЙМВОРКАХ

Т. О. Фединишин, О. О. Партика

Національний університет "Львівська політехніка",
кафедра захисту інформації
E-mail: taras.o.fedynyshyn@lpnu.ua, olha.o.mykhailova@lpnu.ua

Досліджено безпекові ризики мобільних додатків, створених із використанням різних фреймворків розробки, зокрема Xamarin, Cordova, React Native, Flutter та Android Native. Метою дослідження є виявлення ключових вразливостей у коді, конфігураціях та дозволах мобільних додатків, а також оцінка їхньої критичності залежно від обраної технології розробки. У межах дослідження було проведено статичний аналіз 6165 мобільних додатків за допомогою інструменту MobSF, який охоплював аспекти бінарного аналізу, сертифікатів, мережевої безпеки, налаштування Firebase, дозволів і конфігурацій Android Manifest. Отримані результати вказують на значні відмінності у поширеності та критичності вразливостей залежно від фреймворку. Додатки, розроблені на Xamarin, демонструють вищий рівень ризиків у категоріях небезпечних дозволів та бінарних файлів, тоді як Cordova показує найнижчі показники критичних проблем із сертифікатами та експортованими компонентами. Flutter виявляє вразливості в налаштуванні Android Manifest і дозволах, тоді як Android Native має середній рівень безпеки із деякими проблемами мережевої конфігурації. Проведений аналіз підтверджує важливість статичного тестування мобільних додатків на етапах їхньої розробки для мінімізації ризиків. Результати дослідження можуть бути використані для розробки практичних рекомендацій, спрямованих на підвищення безпеки мобільних додатків, та сприяють формуванню більш стійких до кіберзагроз рішень.

Ключові слова: android security, mobile security, mobile privacy, static analysis, android static analysis, mobile static analysis, mobile application security, mobile application data protection, react native security, flutter security.

Вступ

Мобільні додатки є невід'ємною складовою сучасного цифрового середовища, забезпечуючи доступ до інформації, сервісів та послуг у режимі реального часу. З розвитком технологій кількість додатків у магазинах, таких як Google Play Store, постійно зростає, що сприяє ще ширшому їх впровадженню у повсякденне життя. Це створює додатковий попит на якісні рішення у сфері розробки мобільних додатків.

Сучасна розробка додатків охоплює два основних підходи: нативний і кросплатформний. Нативна розробка базується на використанні платформоспецифічних мов програмування, таких як Java чи Kotlin для Android, що забезпечує оптимальну продуктивність і доступ до всіх функцій пристрою. Такий підхід дає змогу створювати високопродуктивні рішення, які ефективно інтегруються з апаратним забезпеченням і надають користувачам винятковий досвід взаємодії.

Кросплатформні фреймворки, зокрема React Native, Xamarin, Cordova та Flutter, дають змогу створювати універсальні додатки [1], які можуть функціонувати на кількох платформах. Завдяки принципу “написав один раз - запускай всюди” ці технології значно скорочують витрати на розробку та спрощують процес оновлення програмного забезпечення. Однак, незважаючи на зручність, такі рішення можуть мати обмеження у продуктивності та безпеці порівняно з нативними додатками.

Мобільні додатки дедалі частіше використовуються для обробки конфіденційної інформації, наприклад, фінансових даних, приватної переписки або геолокації. Це робить питання їхньої безпеки критично важливим. Будь-яка вразливість у додатках може призвести до серйозних наслідків, таких як витік даних, шахрайство або несанкціонований доступ до пристроїв.

Серед ключових аспектів безпеки мобільних додатків потрібно виділити мережеву безпеку, правильність налаштування дозволів, управління сертифікатами та інтеграцію сторонніх сервісів, таких як Firebase. Вразливості в цих сферах можуть створювати можливості для кібератак, компрометації даних чи їхнього несанкціонованого використання. Особливо це стосується кросплатформних додатків, де ризик неправильного налаштування може бути вищим через складність забезпечення сумісності між платформами.

Проведення статичного аналізу мобільних додатків є ефективним підходом для виявлення потенційних загроз і недоліків у коді чи конфігурації додатків. Використання таких інструментів, як MobSF, дає змогу аналізувати широкий спектр аспектів, зокрема файлову структуру, дозволи, мережеві налаштування та конфігурацію AndroidManifest. Це дає можливість виявляти проблемні зони ще на етапі розробки та коригувати їх.

Сьогодні мобільні додатки відіграють ключову роль у багатьох сферах, таких як фінанси, охорона здоров’я, комунікації та управління пристроями Інтернету речей. Їхня безпека стає пріоритетом для розробників, користувачів та бізнесу. Статичний аналіз дає змогу мінімізувати ризики, пов’язані з використанням додатків, і сприяє побудові більш безпечних цифрових екосистем.

Отож дослідження мобільних додатків з точки зору їхньої безпеки є важливим напрямом, який сприяє підвищенню захищеності користувачів та забезпеченню довіри до сучасних технологій. Аналіз вразливостей у розроблених додатках, зокрема тих, які створені на основі кросплатформних фреймворків, є фундаментальним для розуміння сучасних викликів кібербезпеки та пошуку ефективних рішень для їх подолання.

Огляд літературних джерел

Кросплатформенні інструменти пропонують альтернативу розробці нативних мобільних додатків, дозволяючи значно зменшити час і вартість розробки для кількох платформ (Android, iOS) завдяки спільному використанню значної частини коду. Однак такі інструменти можуть створювати ризики для безпеки через додаткові програмні шари та етапи трансляції. У дослідженні [18], проведеному на основі аналізу понад тисячі додатків, створених за допомогою Cordova, виявлено, що низький рівень впровадження механізмів безпеки призводить до значної кількості вразливих додатків.

Автори дослідження [2] виявили, що більшість розробників додатків Android неправильно визначають набір дозволів, що призводить до їх надмірної або недостатньої специфікації та створює ризики для конфіденційності даних користувачів додатків. Автори запропонували систему автоматичного завантаження додатків з Android Market і виконали статичний аналіз 141 372 APK-додатків, створивши карту викликів API, пов’язаних із відповідними дозволами.

Програмний фреймворк Flutter дає змогу створювати кросплатформенні нативні додатки, але їх автоматична конвертація в нативний код несе певні безпекові ризики. Додатки, створені за допомогою Flutter, вразливі до маніпуляцій і динамічних атак, особливо у ворожих середовищах, де зловмисники можуть скористатися вразливостями через реверс-інжиніринг. Автори дослідження [19] рекомендують для захисту додатків впроваджувати комплексні механізми безпеки, такі як виявлення змін під час виконання коду. Незважаючи на популярність Flutter, нехтування заходами безпеки може мати серйозні наслідки для бізнесу.

Результати дослідження [10] показали, що більшість користувачів смартфонів або не звертають уваги на питання конфіденційності, або не розуміють повідомлень про неї, що дозволяє розробникам запитувати надмірну кількість дозволів для своїх додатків. Для вирішення цієї проблеми автори запропонували алгоритм ранжування на основі безпеки, який обчислює рівень нав'язливості додатків, враховуючи запитувані дозволи, отримані системні дії та уподобання користувачів щодо конфіденційності, і демонструє позитивні результати під час його оцінки на мільйоні Android-додатків.

У дослідженні [15] показано, що пандемія та пов'язані з нею локдауни спричинили значне зростання популярності онлайн-гемблінгу, що потребує введення користувачами чутливої інформації, як-от номери кредитних карток та банківські реквізити. Автори розглянули ризики, пов'язані зі завантаженням додатків із категорії гемблінг із неофіційних джерел, які можуть створювати серйозні загрози для безпеки даних користувачів.

У дослідженні [16] подано огляд основних інструментів реверс-інжинірингу Android-додатків, зокрема для декомпіляції APK-файлів, конвертації Java в проміжні мови та аналізу Java-коду, з акцентом на високу точність та ефективність таких інструментів, як App Cloner та Mt Manager.

У відповідь на зростання кількості шкідливих APK-файлів на ринку додатків і активізацію мобільного зловмисного програмного забезпечення у дослідженні [17] проаналізовано механізми безпеки Android та їхні проблеми. Запропоновано автоматизовану систему аналізу APK для потреб судової експертизи, яка характеризується простотою у використанні та вищою точністю розпізнавання порівняно з традиційними системами.

Постановка задачі

Розвиток мобільних додатків за останнє десятиліття супроводжувався значним зростанням кількості загроз інформаційній безпеці. Однією з ключових проблем сучасної розробки є вибір технологій, які дають змогу забезпечити баланс між функціональністю, продуктивністю та безпекою. Сучасні кросплатформні фреймворки, такі як Xamarin, Cordova, React Native, Flutter, а також нативна розробка для Android, активно використовуються для створення мобільних додатків. Водночас їхня безпекова складова залишається недостатньо дослідженою.

Метою цього дослідження є порівняння вразливостей мобільних додатків, створених на базі різних фреймворків, для визначення ризиків, які вони можуть становити для користувачів і розробників.

Актуальність задачі пояснюється стрімким зростанням кількості мобільних додатків та їхнього використання у критичних інформаційних системах, де питання захисту даних є першочерговим. Для реалізації мети необхідно провести детальний статичний аналіз мобільних додатків із використанням інструменту MobSF, який дозволяє ідентифікувати різні типи вразливостей, включаючи проблеми з мережевою безпекою, дозволами, використанням Firebase, а також аналіз конфігурацій Android Manifest. Задачею є не лише виявлення вразливостей, а й визначення їхньої критичності, щоб оцінити потенційний ризик для кінцевих користувачів. Дослідження має на меті також встановити залежність між частотою та типами вразливостей і використаними технологіями розробки. Особливу увагу буде приділено таким аспектам, як інтеграція мережевих сервісів, коректність конфігурацій Firebase та небезпечні дозволи, які можуть відкривати додатки для атак. Окремо буде проаналізовано кількість і типи експортованих сервісів, активностей та провайдерів у додатках.

Результати цього дослідження сприятимуть формуванню кращого розуміння безпекових ризиків різних фреймворків. Це дасть змогу розробникам ухвалювати більш обґрунтовані рішення під час вибору технологій для створення додатків, а також сприятиме розробці рекомендацій щодо покращення безпеки мобільних додатків.

Фреймворки для мобільної розробки

Мобільні додатки сьогодні стали невід'ємною частиною цифрового світу, а їх розробка набуває дедалі більшого поширення завдяки використанню сучасних інструментів і технологій. Основні підходи до розробки мобільних додатків включають нативну розробку, орієнтовану на конкретну платформу, та кросплатформну розробку, яка дає змогу створювати додатки одночасно для кількох платформ. Обидва підходи мають свої переваги та недоліки [3].

Нативна розробка передбачає використання мов програмування, інструментів та API, специфічних для конкретної платформи, наприклад, Java або Kotlin для Android. Вона забезпечує високий рівень продуктивності, доступ до всіх можливостей апаратного забезпечення пристрою та найкращу взаємодію з користувачем. Однак створення окремих додатків для кожної платформи може бути дорогим і трудомістким.

Кросплатформна розробка пропонує альтернативний підхід, який дає змогу створювати додатки за принципом "write once, run anywhere" (написав один раз - запускай всюди). Цей підхід дозволяє знизити витрати та спростити підтримку додатків. Однак кросплатформні додатки можуть поступатися нативним у продуктивності та можливостях інтеграції з апаратним забезпеченням. Серед найпопулярніших кросплатформних фреймворків варто виокремити Xamarin, React Native, Cordova та Flutter.

Xamarin - це фреймворк від Microsoft, який дає змогу створювати мобільні додатки на базі мови програмування C# [6]. Він використовує .NET-платформу, пропонуючи розробникам інструменти для створення додатків із повною підтримкою функцій платформи. Xamarin забезпечує автоматичне управління пам'яттю, збір сміття та підтримку інтеграції з різними платформами.

React Native, створений Facebook [5] у 2015 році, базується на JavaScript [4] і дає змогу створювати додатки для iOS та Android із використанням компонентної архітектури. Цей фреймворк підтримує як нативні модулі, так і JavaScript-рушій, забезпечуючи високу гнучкість і можливість створення складних інтерфейсів.

Cordova - це відкритий фреймворк, який дозволяє розробляти мобільні додатки, використовуючи вебтехнології [7], такі як HTML, CSS та JavaScript. Він забезпечує доступ до нативних функцій пристроїв за допомогою плагінів, що дозволяє запускати додатки на різних платформах без необхідності переписувати код.

Flutter, розроблений Google [8], є інструментом для створення додатків із використанням мови програмування Dart. Його відмінною рисою є багатий набір віджетів, які дають змогу створювати сучасні інтерфейси з високою продуктивністю.

Нативна розробка для Android залишається найпопулярнішим підходом для створення високопродуктивних додатків. Використовуючи Android Studio та мови програмування Java або Kotlin, розробники мають повний доступ до всіх можливостей платформи, що забезпечує найкращий досвід користувача [9].

Отже, вибір між нативною та кросплатформною розробкою залежить від багатьох факторів, включаючи вимоги до продуктивності, ресурсів команди розробників та потреби користувачів. Незважаючи на переваги кросплатформних фреймворків, безпекові аспекти залишаються важливим фактором, який необхідно враховувати під час вибору технологій для розробки.

План експерименту

Метою експерименту було виявлення вразливостей мобільних додатків, створених із використанням різних фреймворків, а також аналіз їхньої поширеності та критичності. Для досягнення цієї мети було обрано 6165 додатків з Google Play Store, які належать до різних категорій і є популярними серед користувачів. Аналіз охоплював додатки, розроблені за допомогою таких фреймворків, як Xamarin, Cordova, React Native, Flutter і Android Native.

Перший етап експерименту передбачав визначення популярності мобільних додатків у різних категоріях. Для цього використовували сервіс Similarweb [11], який дає змогу аналізувати рейтинги додатків у 96 країнах. Цей підхід дав змогу зібрати перелік додатків, які надають широкий спектр функцій і технологій.

Наступним етапом стало завантаження APK-файлів для проведення статичного аналізу. Оскільки Google Play Store не підтримує пряме завантаження APK-файлів, було використано сторонні сервіси, такі як APKCombo [12]. У результаті вдалося отримати 6165 файлів, які відповідали вимогам дослідження.

Для виконання статичного аналізу використовували інструмент MobSF (Mobile Security Framework) [13], що дозволяє ідентифікувати широкий спектр вразливостей у додатках. Було налаштовано з'єднання з базою даних Postgres, щоб автоматизувати обробку отриманих результатів. Аналіз охоплював такі аспекти, як бінарні файли, сертифікати, код, конфігурацію Android Manifest, мережеву безпеку, дозволи та інтеграцію Firebase [14].

Основною задачею аналізу було виявлення вразливостей із високим рівнем критичності. Для кожного додатка оцінювалися небезпечні дозволи, проблеми в налаштуванні мережевих параметрів, експортування служб і активностей, а також наявність потенційно небезпечних файлів, таких як сертифікати або ключі. Додатково проаналізовано конфігурацію Firebase, щоб визначити, чи правильно налаштовані бази даних, доступ до яких може бути відкритим. Отримані результати були оброблені для виявлення шаблонів і залежностей між фреймворками та типами вразливостей. Проведено порівняння кількості критичних проблем у додатках залежно від технології їхньої розробки. Це дало змогу визначити фреймворки, які є найбільш і найменш безпечними з точки зору реалізації.

Під час обробки даних враховувалися як абсолютні показники, так і середні значення вразливостей для кожної групи додатків. Це дало змогу оцінити як поширеність проблем, так і їхню потенційну загрозу для кінцевих користувачів.

Експериментальна процедура дозволила отримати надійні дані для подальшого аналізу безпеки мобільних додатків. Результати дослідження можуть бути використані для розробки рекомендацій із покращення практик розробки, що сприятиме створенню більш захищених додатків у майбутньому.

Результати дослідження

У процесі дослідження було проведено статичний аналіз 6165 мобільних додатків, що дало змогу виявити вразливості та оцінити їхню критичність залежно від фреймворку, використаного для розробки. Аналіз охоплював кілька ключових аспектів, зокрема бінарні файли, сертифікати, конфігурації Android Manifest, дозволи, мережеву безпеку та інтеграцію Firebase.

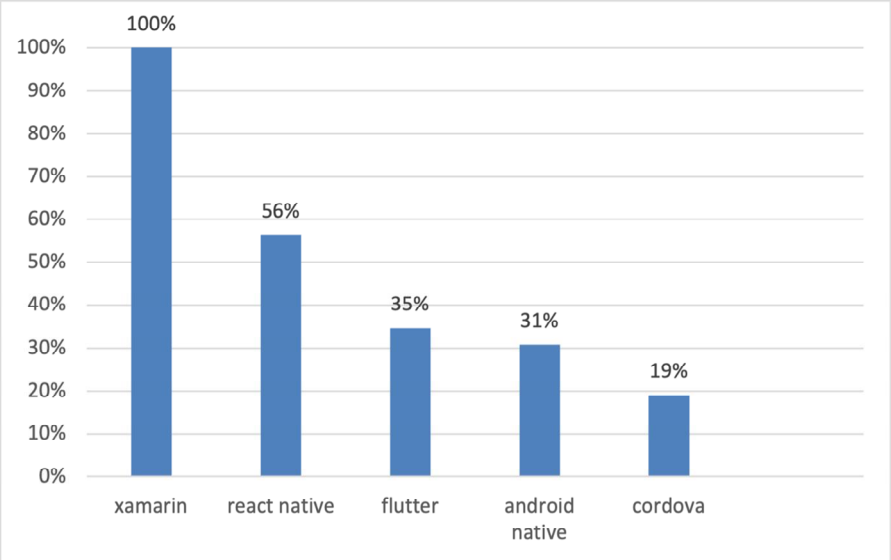

Рис. 1. Відсоток APK-файлів, які містять принаймні одну бінарну складову з проблемою високої критичності

Результати бінарного аналізу на рис. 1 показали, що додатки, створені за допомогою Xamarin, мали найбільший відсоток висококритичних проблем. Усі проаналізовані Xamarin-додатки мали щонайменше одну проблему високої критичності. Водночас додатки, розроблені на Cordova, показали найкращі результати - лише 19 % з них мали подібні вразливості.

Аналіз сертифікатів, зображений на рис. 2, виявив, що Cordova також демонструє найвищу кількість проблем із сертифікатами. Лише 2,63 % додатків, створених за допомогою Flutter, мали серйозні проблеми з сертифікатами, що свідчить про високу якість налаштувань у цьому фреймворку. Натомість додатки на Cordova мали в середньому 14,29 % проблеми з сертифікатами.

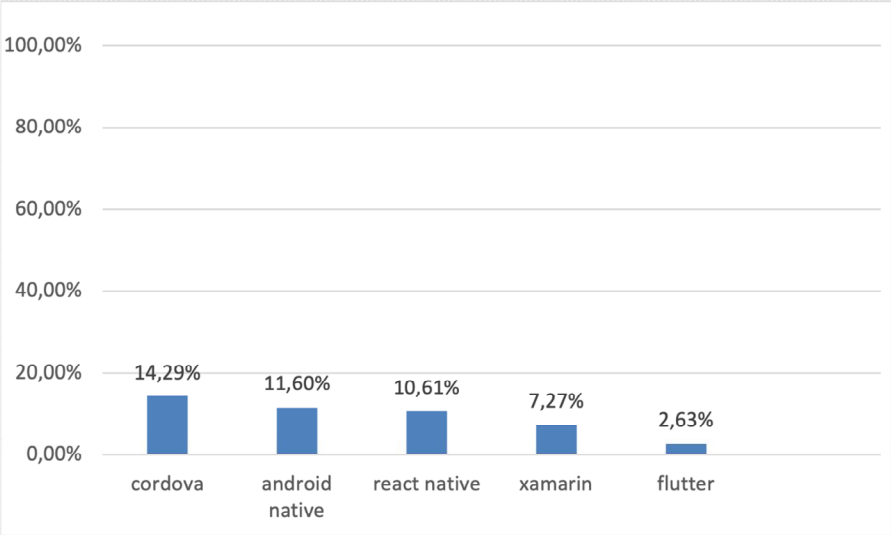

Рис. 2. Відсоток APK-файлів, які містять принаймні одну проблему з сертифікатом високої критичності

Як показано на рис. 3, аналіз коду виявив, що 76 % додатків, створених на Flutter, містять фрагменти з висококритичними проблемами, тоді як Xamarin мав найнижчий показник - лише 13 %. Проте середнє значення кількості таких проблем у Xamarin додатках було меншим, ніж у React Native, що свідчить про різницю в структурі коду залежно від фреймворку – показано на рис. 4.

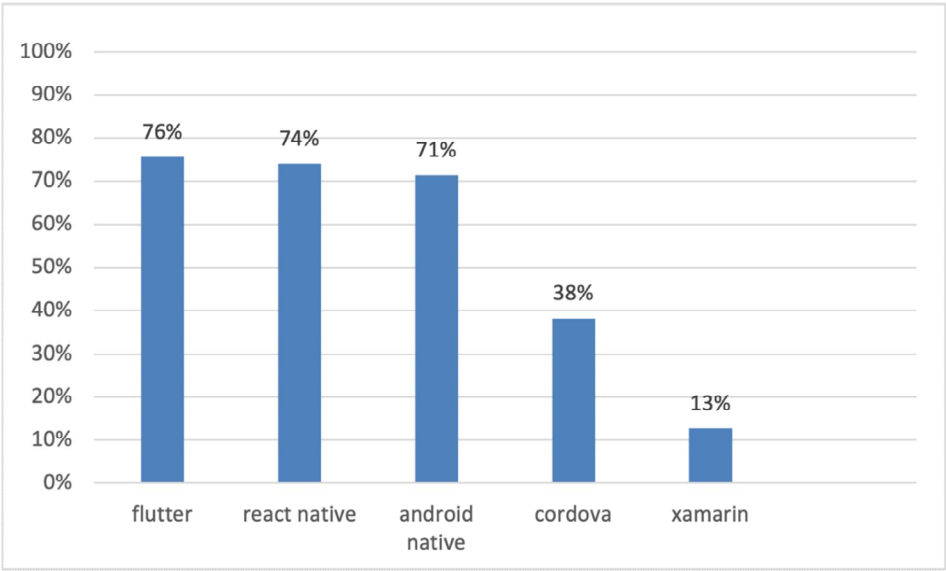

Рис. 3. Відсоток APK-файлів, які містять принаймні один фрагмент коду з проблемою високої критичності

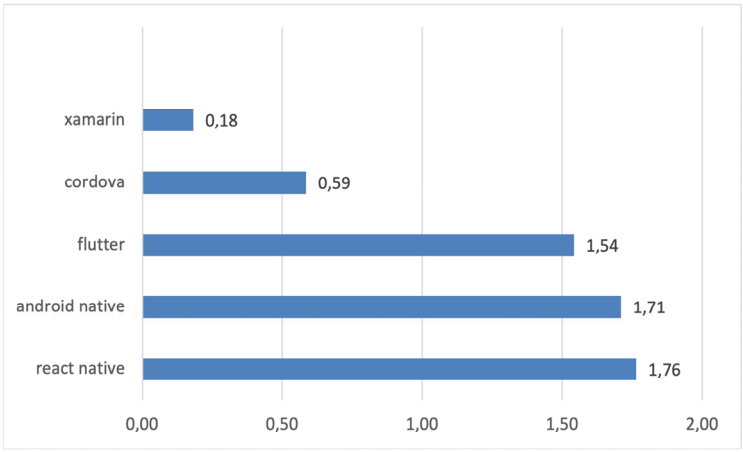

Рис. 4. Середня кількість фрагментів коду з проблемами високої критичності

Дослідження конфігурації AndroidManifest виявило, що 95 % додатків, створених із використанням Flutter, мали щонайменше одну проблему високої критичності в цьому файлі – показано на рис. 5. Найкращі результати показали Xamarin додатки, де цей показник становив 82 %. Як показано на рис. 6, в середньому кожен додаток на Flutter мав 1,99 критичну проблему в Android Manifest, тоді як додатки, створені на Xamarin, лише 0,98.

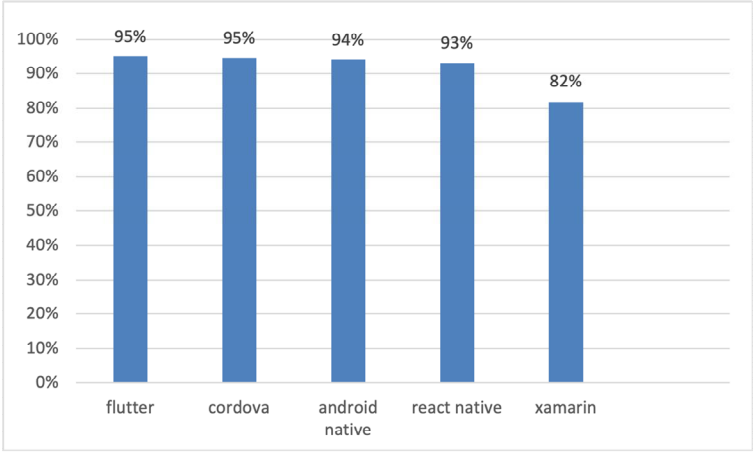

Рис. 5. Відсоток APK-файлів, які містять принаймні одну проблему високої критичності у файлі AndroidManifest.XML

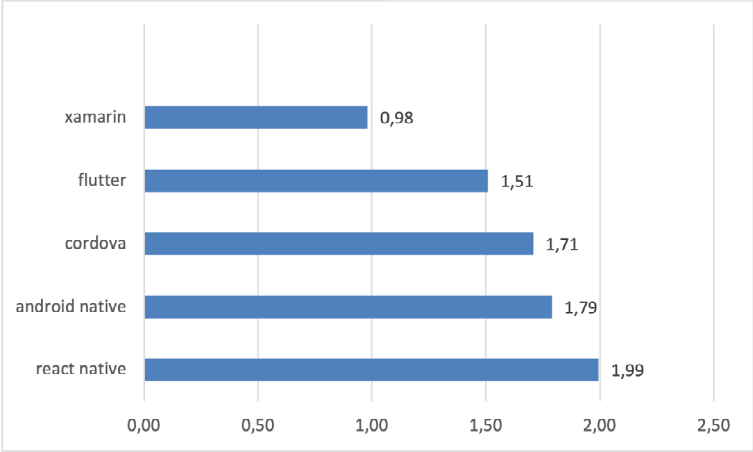

Рис. 6. Середня кількість проблем високої критичності у файлі AndroidManifest.XML

Результат дослідження мережевої безпеки, зображений на рис. 7, показав, що 35 % додатків, створених нативно для Android, мали висококритичні проблеми з конфігурацією мережі. Натомість Xamarin показав найкращі результати в цій категорії - лише 9 % додатків мали проблеми, що свідчить про надійність цього фреймворку у налаштуванні мережевої безпеки.

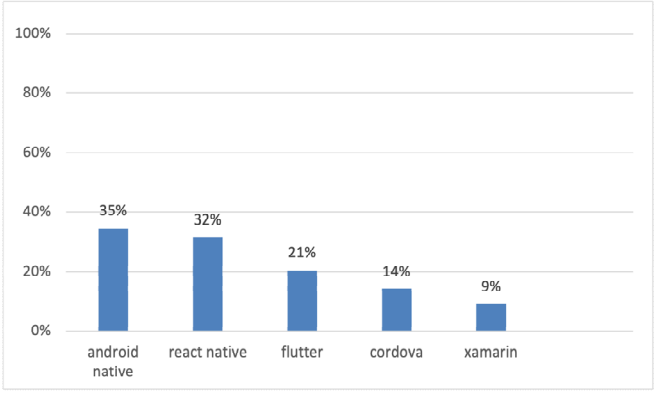

Рис. 7. Відсоток APK-файлів, які містять принаймні одну проблему високої критичності у сфері мережевої безпеки

Аналіз дозволів, показаний на рис. 8, показав, що 100 % додатків на Xamarin запитували щонайменше один небезпечний дозвіл, тоді як Android Native мав найнижчий показник у 91 %. React Native додатки в середньому запитували найбільшу кількість небезпечних дозволів - 6,57 на додаток, що вказує на більший ризик для конфіденційності користувачів – показано на рис. 9.

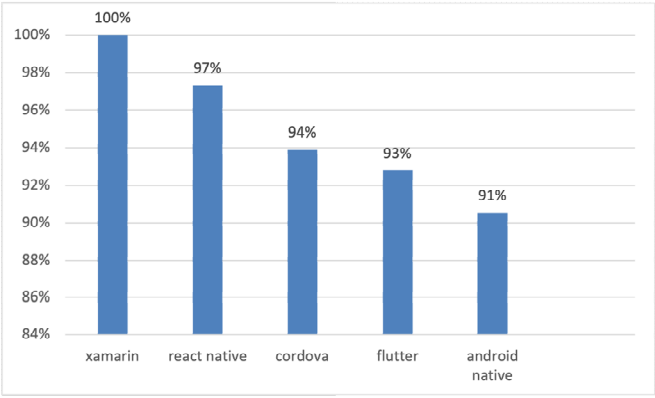

Рис. 8. Відсоток APK-файлів, які містять принаймні один небезпечний дозвіл

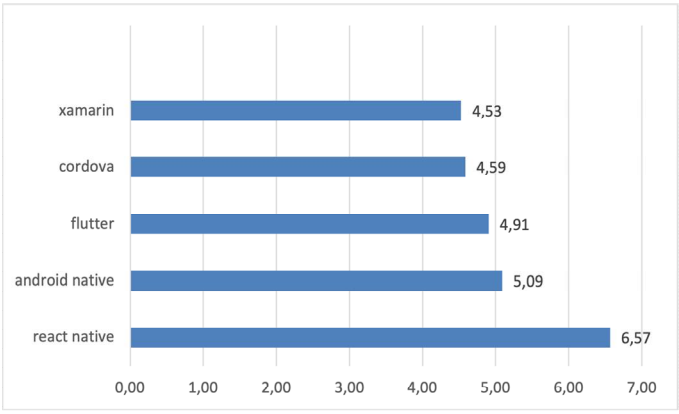

Рис. 9. Середня кількість небезпечних дозволів запитувана додатками

Що стосується інтеграції Firebase, то традиційно створені Android додатки частіше використовували цю платформу, показано на рис. 10. Проте 2,57 % Firebase баз у таких додатках мали відкриті конфігурації, що могло дозволити стороннім особам доступ до даних. Найбезпечнішими в цій категорії виявилися додатки на Xamarin, де проблеми із конфігурацією були мінімальними.

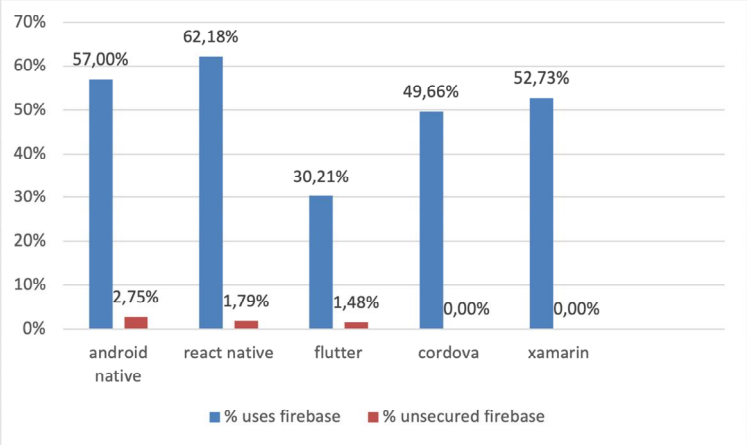

Рис. 10. Відсоток APK-файлів, які мають інтегрований Firebase, та відсоток тих, що налаштовані некоректно

Отримані результати демонструють, що вибір фреймворку значно впливає на безпеку мобільних додатків. Xamarin, хоча й демонструє деякі сильні сторони, має більше проблем із дозволами та бінарними файлами. Cordova та Flutter, навпаки, показали кращі результати у таких аспектах, як сертифікати та конфігурація мереж.

Висновки

У процесі дослідження було проведено комплексний аналіз безпекових аспектів мобільних додатків, розроблених із використанням популярних фреймворків, таких як Xamarin, Cordova, React Native, Flutter та Android Native. Результати статичного аналізу продемонстрували значні відмінності в кількості та критичності вразливостей залежно від обраного підходу до розробки. Це дає змогу зробити висновок, що вибір технології має прямий вплив на рівень захисту мобільного додатку.

Значна кількість потенційних вразливостей була виявлена в додатках, розроблених на Xamarin, що свідчить про потенційні ризики цього фреймворку, особливо в аспектах дозволів і бінарних файлів. Водночас додатки, створені на Cordova, виявилися менш вразливими до багатьох типів загроз, таких як проблеми із сертифікатами та експортованими компонентами. Flutter, хоча й демонструє певні переваги у налаштуванні сертифікатів і файлів, мав більше проблем із конфігурацією Android Manifest і небезпечними дозволами.

Нативна розробка для Android показала середні результати, характеризуючись кращими показниками у використанні Firebase, але маючи проблеми з мережевою безпекою. Аналіз дав змогу виявити важливі закономірності: фреймворки, які використовують складні архітектури, частіше мають критичні проблеми у конфігураціях, тоді як простіші рішення демонструють більшу стійкість до певних типів атак.

Загалом дослідження підтвердило необхідність проведення статичного аналізу на етапах розробки мобільних додатків, незалежно від обраного фреймворку. Отримані результати можуть бути використані для формування рекомендацій розробникам щодо підвищення рівня безпеки додатків, особливо в аспектах налаштування дозволів, конфігурації мережевих параметрів та інтеграції сторонніх сервісів.

Результати дослідження свідчать про важливість системного підходу до аналізу безпеки, який охоплює не лише виявлення вразливостей, а й оцінку їхньої критичності та впливу на користувачів. Надалі це сприятиме створенню більш захищених мобільних рішень, що відповідають сучасним викликам кібербезпеки. Отже, запропонована методологія аналізу та отримані висновки роблять вагомий внесок у вдосконалення практик забезпечення безпеки мобільних додатків.

Список літератури

1. *You Dongliang, Hu Minjie. A Comparative Study of Cross-platform Mobile Application Development. 2021. Doi: https://doi.org/10.1145/3373376.3378534.*
2. *Ortiz-Rodriguez F. et al. An Android App Permission Analysis for User Privacy and Security. Futuristic Trends for Sustainable Development and Sustainable Ecosystems. IGI Global, 2022. Pp. 89- 103. Doi: https://doi.org/10.4018/978-1-6684-4225-8.ch006.*
3. *Aurelius M. Mobile Application Development : Framework with key criteria for choosing native or cross-platform application development, Dissertation, 2020.*
4. *Juho Vepsäläinen, ECMAScript - The journey of a programming language from an idea to a standard, 2023. Doi: https://doi.org/10.48550/arXiv.2305.01373.*
5. *Shevtsiv N., Shvets D., Karabut N. Prospects for Using React Native for Developing Cross-platform Mobile Applications. Central Ukrainian Scientific Bulletin. Technical Sciences. 2019. Col.2(33). Doi: https://doi.org/10.32515/2664-262X.2019.2(33).208-213.*
6. *Xamarin (.NET MAUI), 2024. URL: https://dotnet.microsoft.com/en-us/apps/xamarin.*
7. *Apache Cordova, 2024. URL: https://cordova.apache.org/.*
8. *Singh Prachi. Future of Flutter - An Emerging Technology. Journal of Computer Science and System Software. 2024. 1. 25- 29. Doi: https://doi.org/10.48001/jocsss.2024.1225-29.*
9. *Fedorchenko V., Poliakov A. Analyzing the efficiency of technologies for developing mobile applications for Android OS. Bulletin of Kharkov National Automobile and Highway University, 2022. 81. Doi: https:// doi.org/10.30977/BUL.2219-5548.2022.96.0.81.*
10. *Mohsen F., Abdelhaq H., Bisgin H. Security-centric ranking algorithm and two privacy scores to mitigate intrusive apps. Concurrency Computat Pract Exper. 2022. 34(14):e6571. Doi: https://doi.org/10.1002/cpe.6571.*
11. *Similarweb Digital Intelligence: Unlock Your Digital Growth, 2024. URL: https://www.similarweb.com/.*
12. *APKCombo - #1 APK Downloader, 2024. URL: https://apkcombo.com/.*
13. *Archibong Esther, Stephen Bliss, Asuquo Philip. Analysis of Cybersecurity Vulnerabilities in Mobile Payment Applications. Archives of Advanced Engineering Science. 2024. 1- 12. Doi: https://doi.org/10.47852/ bonviewAAES42022595.*
14. *Firebase. 2024. URL: https://firebase.google.com/.*
15. *Blancaflor E., Pastrana R.LP.J., Sheng M.J.C., Tamayo J.R.D., Umali J.A.M. A Security and Vulnerability Assessment on Android Gambling Applications. Computer and Communication Engineering. CCCE 2023. Communications in Computer and Information Science. 2023. Vol. 1823. Springer, Cham. Doi: https://doi.org/10.1007/978-3-031-35299-7_9.*
16. *Hrushik Raj S., Thejaswini P., Nandi S. Reverse Engineering techniques for Android systems: A Systematic approach, 2023 IEEE Guwahati Subsection Conference, GCON 2023. Doi: https://doi.org/10.1109/GCON58516.2023.10183629.*
17. *Li Y., Liang G., Tang K., Teng Y., Ruan H., Mo Z. (2024). Design and Implementation of an Automated APK Analysis System for Practical Forensics. Proceedings of the 13th International Conference on Computer Engineering and Networks. CENet 2023. Lecture Notes in Electrical Engineering. Vol 1125. Springer, Singapore. Doi: https://doi.org/10.1007/978-981-99-9239-3_36.*
18. *Willocx M., Vossaert J., Naessens V. Security Analysis of Cordova Applications in Google Play. In Proceedings of the 12th International Conference on Availability, Reliability and Security (ARES '17). Association for Computing Machinery, New York, NY, USA, 2017. Article 46, 1–7. Doi: https://doi.org/10.1145/3098954.3103162.*
19. *Rambhia Palash, Shinde Parth, Bamane Kalyan. Securing Flutter Applications: A Comprehensive Study. 2023. 1- 5. Doi: https://doi.org/10.1109/ICCUBEA58933.2023.10392001.*

ANALYSIS OF SECURITY RISKS IN MOBILE APPLICATIONS DEVELOPED WITH CROSS-PLATFORM FRAMEWORKS

T. O. Fedynyshyn, O. O. Partyka

Lviv Polytechnic National University,
Department of Information Protection

The article focuses on the investigation of security risks in mobile applications developed using various frameworks, including Xamarin, Cordova, React Native, Flutter, and Android Native. The purpose of the study is to identify key vulnerabilities in the code, configurations, and permissions of mobile applications and to assess their criticality depending on the chosen development technology. As part of the research, static analysis of 6,165 mobile applications was conducted using the MobSF tool, covering aspects such as binary analysis, certificates, network security, Firebase configuration, permissions, and Android Manifest settings. The results indicate significant differences in the prevalence and criticality of vulnerabilities depending on the framework. Applications developed with Xamarin demonstrate higher risk levels in categories such as dangerous permissions and binary files, while Cordova shows the lowest incidence of critical issues related to certificates and exported components. Flutter reveals vulnerabilities in Android Manifest configurations and permissions, whereas Android Native exhibits a moderate security level with some network configuration issues. The analysis confirms the importance of static testing of mobile applications during their development to minimize risks. The results of the study can be utilized to develop practical recommendations aimed at improving the security of mobile applications and contribute to the creation of solutions that are more resilient to cyber threats.

Keywords: android security, mobile security, mobile privacy, static analysis, android static analysis, mobile static analysis, mobile application security, mobile application data protection, react native security, flutter security.